Distributed Artificial Intelligence And Knowledge Management: Ontologies And Multi-Agent Systems For A Corporate Semantic Web

Fabien Gandon

SCIENTIFIC PHILOSOPHER DOCTORATE THESIS IN INFORMATICS

INRIA and University of Nice - Sophia Antipolis — Doctoral School of Sciences and Technologies of Information and Communication (S.T.I.C.)

DISTRIBUTED ARTIFICIAL INTELLIGENCE AND KNOWLEDGE MANAGEMENT:

ONTOLOGIES AND MULTI-AGENT SYSTEMS FOR A CORPORATE SEMANTIC WEB *

by **Fabien GANDON**

defended Thursday the 7th of November 2002

* Intelligence artificielle distribuée et gestion des connaissances : ontologies et systèmes multi-agents pour un web sémantique organisationnel

THESIS DIRECTOR:	▪ M^ME ROSE DIENG-KUNTZ	(RESEARCH DIRECTOR)
REPORTERS:	▪ M^R JOOST BREUKER	(PROFESSOR)
	▪ M^R LES GASSER	(PROFESSOR)
	▪ M^R JEAN-PAUL HATON	(PROFESSOR)
JURY:	▪ M^R PIERRE BERNHARD	(PROFESSOR - PRESIDENT OF THE JURY)
	▪ M^R JACQUES FERBER	(PROFESSOR)
	▪ M^R HERVÉ KARP	(CONSULTANT)
	▪ M^R GILLES KASSEL	(PROFESSOR)
	▪ M^R AGOSTINO POGGI	(PROFESSOR)

Dedicated

To *my whole family* without whom I would not be where I am, and even worse, I would not be at all.

To my parents *Michel* and *Josette Gandon* for their infallible support and advice to a perpetually absent son.

To my sister *Adeline Gandon* from a brother always on the way to some other place.

To my *friends* for not giving up their friendship to the ghost of myself.

Acknowledgements

To *Rose Dieng-Kuntz*, director of my PhD, for her advice, her support and her trust in me and whithout whom this work would never have been done.

To *Olivier Corby*, for the extremely fruitful technical discussions we had and his constant meticulous work on CORESE.

To *Alain Giboin*, for the inspiring discussions we had on cognitive aspects and his supervision of the ergonomics and evaluation matters.

To the whole ACACIA team for the work we did in common, in particular to *Laurent Berthelot*, *Alexandre Delteil*, *Catherine Faron-Zucker* and *Carolina Medina Ramirez*.

To the assistant of the ACACIA team, *Hortense Hammel* and *Sophie Honnorat* for their advice and help in the day-to-day work and their assistance in the logistic of the PhD.

To all the partners of the CoMMA IST European project, as well as all the trainees involved in it.

To the members of the jury and the reporters of this Ph.D for accepting the extra amount of work.

To *Catherine Barry* and *Régine Loisel* who introduced me to research during my M.Phil.

To the SEMIR department of INRIA, in charge of the computer facilities for its excellent work in maintaining a high quality working environment.

To the librarian department of INRIA, that did a valuable work to provide me with information resources and references vital to my work.

To INRIA and University of Nice Sophia Antipolis for providing me with the means to carry out this work.

To the European Commission that funded the CoMMA IST project

Table of Contents

Introduction

The beginning is the most
important part of the work.
— Plato

Human societies are structured and ruled by organisations. *Organisations can be seen as abstract holonic living entities* composed of individuals and other organisations. The *raison d'être* of these living entities is a set of core activities that answer needs of other organisations or individuals. These core activities are the result of the collective work of the members of the organisation. The global activity relies on organisational structures and infrastructures that are supervised by an organisational management. The management aims at effectively co-ordinating the work of the members in order to obtain a collective achievement of the core organisational activities.

The individual work, be it part of the organisational management or the core activities, requires knowledge. The whole knowledge mobilised by an organisation for its functioning forms an abstract set called organisational knowledge; a lack of organisational knowledge may result in organisational dysfunction.

As the speed of markets is rising and their dimensions tend towards globalisation, reaction time is shortening and competitive pressure is growing; information loss may lead to a missed opportunity. Organisations must react quickly to changes in their domain and in the needs they answer, and even better they must anticipate them. In this context, knowledge is an organisational asset for competitiveness and survival, the importance of which has been growing fast in the last decade. Thus organisational management now explicitly includes the activity of *knowledge management that addresses problems of identification, acquisition, storage, access, diffusion, reuse and maintenance of both internal and external knowledge*.

One approach for managing knowledge in an organisation, is to set up an organisational memory management solution: the *organisational memory* aspect is in charge of ensuring the persistent storage and/or indexing of the organisational knowledge and its *management solution* is in charge of capturing relevant pieces of knowledge and providing the concerned persons with them, both activities being carried out at the appropriate time, with the right level of details and in an adequate format. An organisational memory relies on knowledge resources *i.e.*, documents, people, formalised knowledge (e.g.: software, knowledge bases) and other artefacts in which knowledge has been embedded.

Such memory and its management require methodologies and tools to be operationalised. Resources, such as documents, are information supports therefore, their management can benefit from results in informatics and the assistance of software solutions developed in the field. The work I am presenting here, was carried out during my Ph.D. in Informatics, within ACACIA[1], a multidisciplinary research team of INRIA[2] that aims at offering models, methods and tools for building a corporate memory. My research was applied to the realisation of CoMMA (Corporate Memory Management through Agents) a two-year European IST project.

In this thesis, I shall show that (1) *the semantic webs can provide distributed knowledge spaces for knowledge management;* (2) *ontologies are applicable and effective means for supporting distributed knowledge spaces;* (3) *multi-agent systems are applicable and effective architectures for managing distributed knowledge spaces.* To this end, I devided the plan of this document into four parts which structure a total of thirteen chapters:

The **first part is a guided tour of relevant literature**. It consists of five chapters, the first four chapters analyse the literature relevant to the work and the fifth one positions CoMMA within this survey.

Chapter 1 - Organisational knowledge management. I shall describe the needs for knowledge management and the notion of corporate memory aimed at answering these needs. I shall also introduce three related domains that contribute to building organisational knowledge management solutions: corporate modelling, user modelling and information retrieval systems.

Chapter 2 - Ontology and knowledge modelling. I shall analyse the latest advance in knowledge modelling that may be used for knowledge management, especially focusing on the notion of ontology that is that part of the knowledge model that captures the semantics of primitives used to make formal assertions about the application domain of the knowledge-based solution. The chapter is divided in two large sections respectively addressing the nature and the design of an ontology.

Chapter 3 - Structured and semantic web. More and more organisations rely on intranets and internal corporate webs to implement a corporate memory solution. In the current Web technology, information resources are machine readable, but not machine understandable. To improve exploitation and management mechanisms of a web requires the introduction of formal knowledge; in the semantic web vision of the W3C[3], this takes the form of semantic annotations about the information resources. After a survey of the pioneer projects that prefigured the semantic web, I shall detail the current foundations layed down by the W3C, and summarise the on-going and future trends.

Chapter 4 - Distributed artificial intelligence. Artificial intelligence studies artificial entities, that I shall call agents, reproducing individual intelligent behaviours. In application fields where the elements of the problem are scattered, distributed artificial intelligence tries to build artificial societies of agents to propose adequate solutions; a semantic web is a distributed landscape that naturally calls for this kind of paradigm. Therefore, I shall introduce agents and multi-agent systems and survey design methodologies. I shall focus, in particular, on the multi-agent information systems.

[1] Research team studying 'Methods, models and tools for Knowledge Management' - http://www-sop.inria.fr/acacia/
[2] French National Research Institute in Informatics and Automation - http://www.inria.fr
[3] World Wide Web Consortium - http://www.w3.org/

Chapter 5 - Research context and positioning. Based on the previous surveys, this chapter will identify the requirements of the application scenarios of CoMMA and show that the requirements can be matched with solutions proposed in knowledge modelling, semantic web and distributed artificial intelligence. I shall describe and justify the overall solution envisioned and the implementation choices, and I shall also position CoMMA in the state of the art.

The **second part concerns ontology & annotated corporate memory**. It consists of three chapters presenting a vision of the corporate memory as a corporate semantic web as well as the approach followed in CoMMA to build the underlying ontology and the result of this approach *i.e.,* the ontology O'CoMMA.

Chapter 6 - Corporate semantic web. It is a short introduction to the vision of a corporate memory as a corporate semantic web providing an annotated world for information agents. I shall present the motivations, the structure and the tools of this model-based documentary memory.

Chapter 7 - Engineering the O'CoMMA ontology. The ontology plays a seminal role in the vision of a corporate semantic web. This chapter explains step-by-step how the ontology O'CoMMA was built, following a scenario-based design rationale to go from the informal descriptions of requirements and specifications to a formal ontology. I shall discuss the tools designed and adopted as well as the influences and motivations that drove my choices.

Chapter 8 - The ontology O'CoMMA. The ontology O'CoMMA is the result produced by the methodology adopted in CoMMA and provides the semantic grounding of a solution. I shall describe the O'CoMMA ontology and different aspects of the vocabulary organised in three layers and containing an abstract top layer, a part dedicated to the organisational structure modelling, a part dedicated to the description of people, and a part dealing with domain topics. I shall also discuss some extensions and appropriation experiences with end-users.

The **third part focuses on a multi-agent system for memory management**. It consists of two chapters respectively presenting the design rationale of the whole CoMMA system and the specific design of the mechanisms handling annotation distribution.

Chapter 9 - Design rationale of the multi-agent architecture. I present the design rationale that was followed in CoMMA to obtain the multi-agent architecture supporting the envisaged corporate memory management scenarios. I shall explain every main stage of the organisational top-down analysis of functionalities and describe the characteristics and documentation of roles and protocols supporting each agent sub-society, finally coming down to their implementation into agent behaviours.

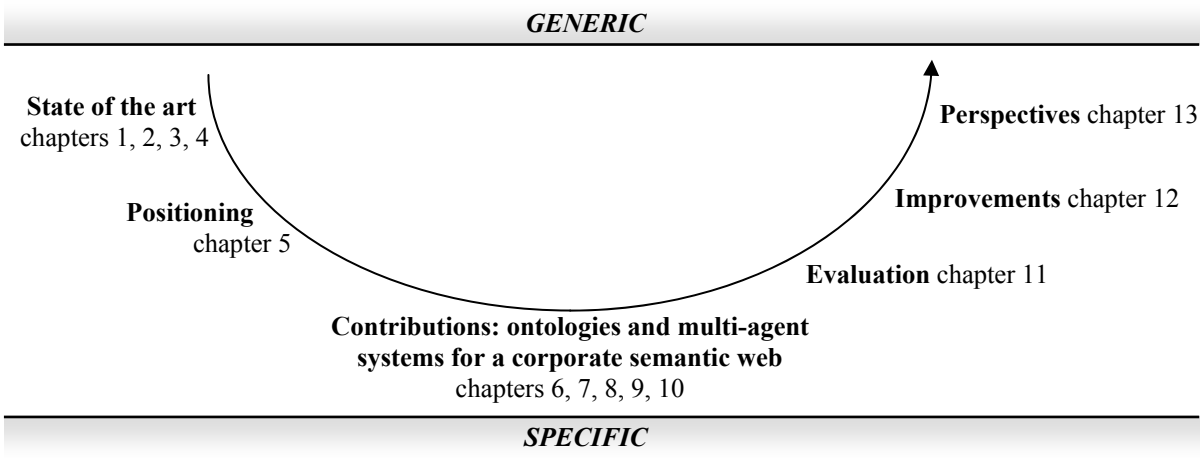
Chapter 10 - Handling annotation distribution. I detail how some aspects of the conceptual foundations of the semantic web can support a multi-agent system in allocating and retrieving semantic annotations in a distributed corporate memory. I shall show how our agents exploit the semantic structures and the ontology, when deciding where to allocate new annotations and when resolving queries over distributed bases of annotations.

The **fourth part gives the lessons learned and the perspectives**. It consists of three chapters, the first one discussing the evaluation and return on experience of the CoMMA project, the second one describing short-term improvements and current work, and the last one proposing some long-term perspectives.

Chapter 11 - Evaluation and return on experience. This chapter gives the implementation status of CoMMA and the trial process to get a feedback from the end-users and carry-out the official evaluation of CoMMA. I shall present the criteria used, the trial feeback, and the lessons learned.

Chapter 12 - Short-term improvements. This chapter gives beginnings of answers to some of the problems raised by the evaluation. The ideas presented are extensions of the work done in CoMMA, some of them already started to be implemented and tested, while others are shallow specifications of possible improvements.

Chapter 13 - long-term perspectives. I provide considerations on extensions that could be imagined to go towards a complete real-world solution and I give long-term perspectives that can be considered as my future research interests.



As shown on the above schema, the reading follows a classical U-shape progress from generic concerns to specific contributions, back and forth. I included as much information material as I found necessary to provide this document with a reasonable autonomy for a linear reading. I sincerely hope that the readers of these pages will find information here to build new knowledge.

— Fabien Gandon

Part - I

Guided tour of relevant literature

He who cannot draw on 3000 years of culture is living from hand to mouth.
— Goethe

1 Organisational knowledge management

The objectives of knowledge management are usually structured by three key issues: *Capitalise* (*i.e.,* know where you are and you come from to know where you go), *Share* (*i.e.,* switch from individual to collective intelligence), *Create* (*i.e.,* anticipate and innovate to survive) [Ermine, 2000]. How to improve identification, acquisition, storage, access, diffusion and reuse and maintenance of both internal and external knowledge in an organisation? This is the question at the heart knowledge management. One approach for managing knowledge in an organisation is the set-up of an organisational memory. Information or knowledge management systems, and in particular a corporate memory, should provide the concerned persons with the relevant pieces of knowledge or information at the appropriate time, with the right level of details and the adequate format of presentation. I shall discuss the notion of organisational memory in the second sub-section.

Then, I shall give an introduction to three domains that are now closely linked to organisational knowledge management

- *corporate modelling*: the need for capturing an explicit view of the organisation that can support the knowledge management solution led to include organisational models as parts of the organisational memory.
- *user modelling*: the need for capturing the profile of the organisation's members as future users, led user modelling field to find a natural application in information and knowledge management for organisations.
- *information retrieval systems*: accessing the right information at the right moment may enable people to get the knowledge to take the good decision at the right time; the retrieval is one of the main issues of information systems and different techniques have been developed that can be use to support knowledge management.

1.1 Needs for knowledge management

1.1.1 Needs due to the organisational activity

Core business resource: in 1994, Drucker already noticed how knowledge was a primary resource of the society and he believed that the implications of this shift would prove increasingly significant for organisations (commercial companies, public institutions, etc.) [Drucker, 1994]. The past decade with its emergence of knowledge workers and knowledge intensive organisations proved he was right. Knowledge is now considered as a capital which has an economic value; it is a strategic resource for increasing productivity; it is a stability factor in a unstable and dynamic competitive environment and it is a decisive competitive advantage [Ermine, 2000]. This encompasses also protecting intellectual property of this capital of knowledge; as soon as a resource becomes vital to a business, security and protection concern arise. Knowledge management thus is also concerned with the problem of securing new and existing knowledge and storing it to make it persistent over time. Also, as knowledge production becomes more critical, managers will need to do it more reflectively [Seely Brown and Duguid, 1998]. This means there is a requirement for methods and tools to assist the management processes.

Transversal resource: knowledge concerns R&D, management (strategy, quality, etc.), production (data management, documentation, workflows, know-how, etc.), human resources (competencies, training, etc.) [Ermine, 1998]. Moreover, knowledge is often needed far from were it was created (in time and space): e.g. the production data may be useless to the worker of the production line, but they are vital to the analyst of the management group that will mine them and correlate them to build key indicators and take strategic decisions; likewise the people of the workshop may be extremely interested in the knowledge discovered by technology monitory group or lessons learned in other workshops. As summarised in [Nagendra Prasad and Plaza, 1996], timely availability of relevant information from resources accessible to an organisation can lead to more informed decisions on the part of individuals, thereby promoting the effectiveness and viability of decentralised decision making. Decision making can be on the assembly line to solve a problem (e.g. past project knowledge) or at the management board (e.g. manage large amount of data and information and mine them to extract key indicators that will drive strategic decisions). Sharing the activity in a group also implies to share the knowledge involved in the activity. To survive, the organisation needs to share knowledge among its members and create and collect new knowledge to anticipate the future. [Abecker *et al.*, 1998]

1.1.2 Needs due to external organisational factors

Competitive advantage: "in an economy where the only certainty is uncertainty, the sure source of lasting competitive advantage is knowledge" [Nonaka and Takeuchi, 1995]. Globalisation is leading to strong international competition which is expressed in rapid market developments and short lifecycles for products and services [Weggeman, 1996]. Therefore, it is vital to maintain knowledge generation as a competitive advantage for knowledge intensive organisations: this is critical, for instance, in trying to reduce the time to market, costs, wastes, etc. An organisation is a group of people who work together in a structured way for a shared purpose. Their activity will mobilise knowledge about the subject of their work. This knowledge has been obtained by experience or study and will drive the behaviour of the people and therefore, determine the actions and reactions of the organisation as a whole. Through organisational activities, additional knowledge will be collected and generated; if it is not stored, indexed and reactivated when needed, then it is lost. In a global market where information flows at the speed of Internet, organisational memory loss is dangerous;

an efficient memory is becoming a vital asset for an organisation especially if its competitors also acknowledge that fact.

Globalisation: globalisation was terribly boosted by the evolution of information technology. As described by Fukuda in 1995, it started with the use of main-frame computers for business information processing in the 60s and 70s, then it spread through the organisations with the micro computers even starting to enter homes in the 80s and it definitively impacted the whole society from the mid 80s by introducing networks and their world-wide webs of interconnected people and organisations. "Now that they can communicate interactively and share information, they come to know that they are affected or regulated by the information power". [Fukuda, 1995]

1.1.3 Needs due to internal organisational factors

Turnover: as the story goes, if NASA was to send men to the moon again, it would have to start from scratch, having lost not the data, but the human expertise that made possible the event of the 13th of July 1969. An organisation's knowledge walks out of the door every night and it might never come back[1]. Employees and knowledge gained during their engagement are often lost in the dynamic business environment [Dzbor *et al.*, 2000]. The turn over in artefacts (e.g. software evolution, products evolution) and in members of the organisation (retirement, internal and external mobility, etc.) are major breaks in knowledge capitalisation.

Need for awareness: members of an organisation are often unaware of critical resources that remain hidden in the vast repositories. Most of the knowledge is thus forgotten in a relatively short time after it was invented [Dzbor *et al.*, 2000]. Competitive pressure requires quick and effective reactions to the ever changing market situations. The gap between the evolving and continuously changing collective information and knowledge resources of an organisation and the employees' awareness of the existence of such resources and of their changes can lead to loss in productivity. [Nagendra Prasad and Plaza, 1996]. Too often one part of an organisation repeats work of another part simply because it is impossible to keep track of, and make use of, knowledge in other parts. Organisations need to know what their corporate knowledge assets are and how to manage and make use of these assets to capitalise on them. They are realising how important it is to "know what they know" and be able to make maximum use of this knowledge [Macintosh, 1994].

Project oriented management: this new way of management aims at promoting organisations as learning systems and avoiding repeating the same mistakes. Information about past projects - protocols, design specifications, documentation of experiences: both failures and successes, alternatives explored - can all serve as stimulants for learning, leading to "expertise transfer" and "cross-project fertilisations" within and across organisations. [Nagendra Prasad and Plaza, 1996]

Internationalisation and geographic dispersion: The other side of the coin of globalisation is the expansion of companies to global trusts. As companies expand internationally, geographic barriers can affect knowledge exchange and prevent easy access to information [O'Leary, 1998]. It thus amplifies the already existing problem of knowledge circulation and knowledge distribution between artefacts and humans dispersed in the world. Moreover, the internationalisation raises the problem of making knowledge cross-cultural and language boundaries.

[1] Kevin Abley (Cap Gemini) in OR Society Conference on Knowledge Management, London, Nov 1998.

1.1.4 Needs due to the nature of information and knowledge

Information overload: with more and more data online (databases, internal web, data warehouse, Internet etc.), "we are drowning in information, but starved of knowledge"[1]. The question raised are: what is relevant and what is not? what are the relevance criteria? how do they evolve? can we predict / anticipate future needs? The amount of available information resources augment exponentially, overloading knowledge-workers; too much information hampers decision making just as much as insufficient knowledge does.

Information heterogeneity: the knowledge assets reside in many different places such as: databases, knowledge bases, filing cabinets and people heads. They are distributed across the enterprise [Macintosh, 1994] and the intranet and internet phenomena amplifies that problem. The dispersed, fragmented and diverse sources augment the cognitive overload of knowledge workers that need integrated homogeneous views to interpret them into 'enactable' knowledge.

For all these reasons, one of the rare consensus in the knowledge management domain is that knowledge is now perceived as an organisational and production asset, a valuable patrimony to be managed and thus there is a need for tools and methods assisting this management.

One approach for managing knowledge in an organisation is the set-up of an organisational memory. Information or knowledge management systems, and in particular a corporate memory, should provide the concerned persons with the relevant pieces of knowledge or information at the appropriate time with the right level of details and the adequate format of presentation. I shall discuss the notion of organisational memory in the following sub-section.

[1] John Naisbitt

1.2 Organisational memories

I use 'corporate memory' or 'organisational memory' as synonyms; the different concepts that are sometimes denoted by these terms will be explored in the section about typology of memories.

Organisations are societies of beings: if these beings have an individual memory, the organisation will naturally develop a collective memory being at the very least the sum of these memories and often much more. From that perspective the question here is what is knowledge management with regard to corporate memories? For [Ackerman and Halverson, 1998] it is important to consider an organisational memory as both an object and a process: it holds its state and it is embedded in many organisational and individual processes. I stress the difference here between this static aspect (the knowledge captured) and dynamic aspect of the memory (the ability to memorise and remember) because I believe they are unconditionally present in a complete solution.

A memory without an intelligence (singular or plural) in charge of it is destined for obsolescence, and an intelligence without a memory (internal or external) is destined for stagnation.

In fact, I see three aspects that are used to define what is a corporate memory: the memory content (what) *i.e.,* the nature of knowledge; the memory form (where) *i.e.,* the storage support; the memory working (how) *i.e.,* the system managing knowledge. Here are some elements of definition found in the literature for each one of these facets:

- *content*: it contains the organisational experience acquired by employees and related to the work they carry out [Pomian, 1996]. It is a repository of knowledge and know-how from a set of individuals working in a particular firm [Euzenat, 1996]. It captures a company's accumulated know-how and other knowledge assets [Kuhn and Abecker, 1997]. It consists of the total sum of the information and knowledge resources within an organisation. Such resources are typically distributed and are characterised by multiplicity and diversity: company databases, machine-readable texts, documentation resources and reports, product requirements, design rationale etc. [Nagendra Prasad and Plaza, 1996] The corporate knowledge is the whole know-how of the company, *i.e.,* its business processes, its procedures, its policies (mission, rules, norms) and its data [Gerbé, 2000]. It is an explicit, disembodied, persistent representation of the knowledge and information in an organisation [Van Heijst *et al.*, 1996]. It preserves the reasoning and knowledge with their diversity and contradictions in order to reuse them later [Pomian, 1996]
- *form*: [Kuhn and Abecker, 1997] characterised corporate memory as a comprehensive computer system. In other approaches organisational memories take the form of an efficient librarian solution based on document management systems. Finally, it can rely on a human resources management policy relying on human as 'knowledge containers'.
- *working*: a corporate memory makes knowledge assets available to enhance the efficiency and effectiveness of knowledge-intensive work processes [Kuhn and Abecker, 1997]. It is a system that enables the integration of dispersed and unstructured organisational knowledge by enhancing its access, dissemination and reuse among an organisation’s members and information systems [von Krogh, 1998]. The main function of a corporate memory is that it should enhance the learning capacity of an organisation [Van Heijst *et al.*, 1996]

My colleagues proposed their own definition of a corporate memory: "explicit, disembodied, persistent representation of knowledge and information in an organisation, in order to facilitate its access and reuse by members of the organisation, for their tasks" [Dieng *et al.*, 2001]. While I find it a good digest of the different definitions, I do not think that knowledge must be disembodied and explicitly represented. I believe that a complete corporate memory solution (*i.e.,* memory & management system) should allow for the indexing of external sources (may be embodied in a person or hold in another memory). This is because, contrary to its human counterpart, the organisational memory tends not to be centralised, localised and enclosed within a physical boundary, but distributed, diffuse and heterogeneous. Moreover, just like its human counterpart, it does not memorise everything it encounters: sometimes it is better to reference/index an external resource rather than duplicate it so that we can still consult it, but do not really have to memorise and maintain it. The reasons may be because it is cost-effective, because it is not feasible (copyright, amount of data, difficult formalisation, etc.), because it is ever changing and volatile, because the maintenance is out of our expertise, etc.

Therefore, I would suggest an extended definition:

An organisational memory is an explicit, disembodied, persistent *representation and indexing of knowledge and information or their sources* in an organisation, in order to facilitate its access, share and reuse by members of the organisation, for their individual and collective tasks.

The desire of management and control over knowledge, contrasts with its fluid, dispersed, intangible, subjective and sometime tacit original nature. How can we manage something we even have problem to define? Therefore, there has been an extensive effort to analyse this knowledge asset and characterise it. I shall give an overview of this discussion in a first following sub-part.

Management consists of activities of control and supervision over other activities. In the case of knowledge management, what are these activities? what does management of the collective knowledge of an organisation consist in? In the second following sub-part, I shall identify and describe the different activities and processes involved.

Finally, I shall give a typology of the memories that have been envisaged so far. It could also be seen as a typology of the different facets of a complete solution trying to integrate the different types of knowledge that must flow and interact, for an organisation to live.

1.2.1 Nature of organisational knowledge

According to [Weggeman, 1996], knowledge and information are primitive terms *i.e.,* terms that are understood although they cannot be accurately defined and the meaning of which lies in the correct use of the concept that can only be learnt by practising the use.

If we take the ladder of understanding (Figure 1) used in the librarian community, we can propose some definitions of data, information and knowledge. These definitions can be built bottom-up or top-down.

Starting from the bottom, I have merged my definitions with the disjunctive definition of knowledge information and data of [Weggeman, 1996] and [Fukuda, 1995]:

- *Data is a perception, a signal, a sign* or a quantum of interaction (e.g. '40' or 'T' are data). Data is symbolic representation of numbers, fact, quantities; an item of data is what a natural or artificial sensor indicates about a variable. [Weggeman, 1996]. Data are made up of symbols and figures which reflect a perception of the experiential world [Fukuda, 1995].
- *Information is data structured according to a convention* (e.g. T=40°). Information is the result of the comparison of data which are situationally structured in order to arrive at a message that is significant in a given context [Weggeman, 1996]. Information is obtained from data which have been given a significance and selected as useful [Fukuda, 1995].
- *Knowledge is information with a context and value* that make it usable (e.g. "the patient of the room 313 of the Hospital of Orleans has a temperature T=40°"). Knowledge is what places someone in the position to perform a particular task by selecting, interpreting and evaluation information depending on the context. [Weggeman, 1996] This is why knowledge empowers people. Knowledge is systematised information which I understand as the information being arranged according to a definite plan or scheme *i.e.,* existing knowledge. Knowledge is an information which was interpreted (*i.e.,* the intended meaning of which was decided) in context and which meaning was articulated with already acquired knowledge [Fukuda, 1995].

Wisdom can be defined as timeless knowledge and [Fukuda, 1995] adds an intermediary step for theory which he defines as generalised knowledge.

Interestingly, I found no trace in literature of top-down definitions while in presenting and re-presenting cycles for individual and collective manipulation of knowledge, we are going up the ladder to peruse and going down the ladder to communicate. Going down the ladder, I would propose:

- *Knowledge is understanding* of a subject which has been obtained by experience or study.
- *Information can be defined as knowledge expressed according to a convention* or knowledge in transit; The Latin root *informare* means "to give form to". In an interview, Nonaka explained that information is a flow of a messages, and knowledge is a stock created by accumulating information; thus, information is a necessary medium or material for eliciting and constructing knowledge. The second difference he made was that information is something passive while knowledge comes from belief, so it is more proactive.
- *Data can be defined as the basic element of information coding*.

Figure 1 Ladder of understanding

I now look at the different categories and characteristics of knowledge that were identified and used in the literature:

- *Formal knowledge vs. informal knowledge*: In its natural state, the knowledge context includes a situation of interpretation and an actor interpreting the information. However, when we envisage an explicit, disembodied and persistent representation of knowledge, it means that it is an artificial knowledge (the counterpart of an artificial intelligence), or more commonly called formalised knowledge where information and context are captured in a symbolic system and its attached model that respectively enable automatic processing and unambiguous interpretation of results and manipulations. This view which was developed in artificial intelligence is close to the wishful thinking of knowledge management according to which knowledge could be an artificial resource that only has value within an appropriate context, that can be devalued and revalued, that is inexhaustible and used, but not consumed. Unfortunately a lot of knowledge comes with a container (human, news letter, etc.) that does not comply to this. The first axiom of [Euzenat, 1996] is that "knowledge must be stated as formally as possible (...) However, not everything can and must be formalised and even if it were, the formal systems could suffer from serious limitations (complexity or incompleteness). " It followed by a second axiom stating that "it must be possible to wrap up a skeleton of formal knowledge with informal flesh made of text, pictures, animation, etc. Thus, knowledge which has not yet reached a formal state, comments about the production of knowledge or informal explanations can be tied to the formal corpora." [Euzenat, 1996]. These axioms acknowledge the fact that in a memory there will be different degrees of knowledge formalisation and that the management system will have to deal with that aspect.
- *Cognitivist perspective vs. constructionist perspective* [von Krogh, 1998]**:** The cognitivist perspective suggests that knowledge consists of "representations of the world that consist of a number of objects or events". The constructionist perspective suggests that knowledge is "an act of construction or creation". Also, Polanyi [Polanyi, 1966] regards knowledge as both static "knowledge" and dynamic "knowing".
- *Tacit knowledge vs. explicit knowledge* [Nonaka, 1994] drawing on [Polanyi, 1966]: tacit knowledge is mental (mental schemata, beliefs, images, personal points of view and perspectives, concrete know-how e.g. reflexes, personal experience, etc.). Tacit knowledge is personal, context-specific, subjective and experience based knowledge, and therefore, hard to formalise and communicate. It also includes cognitive skills such as intuition as well as technical skills such as craft and know-how. Explicit knowledge, on the other hand, is formalised, coded in a language natural (French, English, etc.) or artificial (UML, mathematics, etc.) and can be transmitted. It is objective and rational knowledge that can be expressed in words, sentences, numbers or formulas. It includes theoretical approaches, problem solving, manuals and databases. As explicit knowledge is visible, it was the first to be managed or, at least, to be archived. [Nonaka and Takeuchi, 1995] pointed out that tacit knowledge is also important and raises additional problems; it was illustrated by these companies having to hire back their fired or retired seniors because they could not be replaced by the newcomers having only the explicit knowledge of their educational background and none of the tacit knowledge that was vital to run the business. Tacit knowledge and explicit knowledge are not totally separate, but mutually complementary entities. Without experience, we cannot truly understand. But unless we try to convert tacit knowledge to explicit knowledge, we cannot reflect upon it and share it in the whole organisational (except through mentoring situations *i.e.,* master-apprentice co-working to ensure transfer of know how).
- *Tacit knowledge vs. focal knowledge*: with references to Polanyi, [Sveiby, 1997] discusses tacit *vs.* focal knowledge. In each activity, there are two dimensions of knowledge, which are mutually exclusive: the focal knowledge about the object or phenomenon that is in focus; the tacit knowledge that is used as a tool to handle or improve what is in focus. The focal and tacit dimensions are complementary. The tacit knowledge functions as a background knowledge which assists in accomplishing a task which is in focus. What is tacit varies from one situation to another.

- *Hard knowledge vs. soft knowledge*: [Kimble *et al.*, 2001] propose hard and soft knowledge as being two parts of a duality. That is all knowledge is to some degree both hard and soft. Harder aspects of knowledge are those aspects that are more formalised and that can be structured, articulated and thus 'captured'. Soft aspects of knowledge on the other hand are the more subtle, implicit and not so easily articulated.
- *Competencies* (know-how responsible and validated) / *theoretical knowledge* "know-that" / *procedural knowledge* / *procedural know-how* / *empirical know-how* / *social know-how* [Le Bortef, 1994].
- *Declarative knowledge* (fact, results, generalities, etc.) *vs. procedural knowledge* (know-how, process, expertise, etc.)
- *Know-how vs. skills*: [Grundstein, 1995; Grundstein and Barthès, 1996] distinguish on the one hand, know-how (ability to design, build, sell and support products and services) and on the other hand, individual and collective skills (ability to act, adapt and evolve).
- *Know-what vs. know-how* [Seely Brown and Duguid, 1998]: the organisational knowledge that constitutes "core competency" requires know-what and know-how. The know-what is explicit knowledge which may be shared by several persons. The "know-how" is the particular ability to put know-what into practice[1]. While both work together, they circulate separately. Know-what circulates with relative ease and is consequently hard to protect. Know-how is embedded in work practice and is *sui generis*[2] and thus relatively easy to protect. Conversely, however, it can be hard to spread, co-ordinate, benchmark, or change. Know-how is a disposition, brought out in practice. Thus, know-how is critical in making knowledge actionable and operational.
- *Company knowledge vs. corporate knowledge* [Grunstein and Barthès, 1996]: company knowledge is technical knowledge used inside the company, its business units, departments, subsidiaries (knowledge needed everyday by the company employees); corporate knowledge is strategic knowledge used by the management at a corporate level. (knowledge about the company)
- *Distributed knowledge vs. centralised knowledge*: [Seely Brown and Duguid, 1998] The distribution of knowledge in an organisation, or in society as a whole, reflects the social division of labour. As Adam Smith insightfully explained, the division of labour is a great source of dynamism and efficiency. Specialised groups are capable of producing highly specialised knowledge. The tasks undertaken by communities of practice develop particular, local, and highly specialised knowledge within the community. Hierarchical divisions of labour often distinguish thinkers from doers, mental from manual labour, strategy (the knowledge required at the top of a hierarchy) from tactics (the knowledge used at the bottom). Above all, a mental-manual division predisposes organisations to ignore a central asset, the value of the know-how created throughout all its parts.
- *Descriptive knowledge vs. deductive knowledge vs. documentary knowledge* [Pomian, 1996]: descriptive knowledge is about history of the organisation, chronological events, actors, and domain of activity. Deductive knowledge is about the reasoning (diagnostic, planning, design, etc.) and their justification (considered aspects, characteristics, facts, results, etc.). It is equivalent to what some may call logic and rationale. Finally, documentary knowledge is knowledge about documents (types, use, access, context, etc.), their nature (report, news, etc.), their content (a primary document contains raw / original data; a secondary document contains identification / analysis of primary documents; a tertiary contains synthesis of primary or secondary documents) and if they are dead (read-only document, frozen once for all) or living (changing its content, etc.). J. Pomian rightly insists on the importance of the interactions between these types of knowledge. One can also find finer-grained descriptions: here the static knowledge was decomposed into descriptive and documentary knowledge, but the dynamic aspect could be decomposed into rationale (plan, diagnostic, etc.) *vs.* heuristic (rule of the thumb, etc.) *vs.* theories *vs.* cases *vs.* (best) practices.

[1] distinction between know-what and know-how and the notion of "dispositional knowledge" comes from Gilbert Ryle, The Concept of Mind (London: Hutchinson, 1954)
[2] constituting a class of its own; being the only example of its kind

- *Tangible knowledge vs. intangible knowledge* [Grunstein and Barthès, 1996]: tangible assets are data, document, etc. while intangible assets are abilities, talents, personal experience, etc. Intangible assets require an elicitation to become tangible before they can participate to a materialised corporate memory
- *Technical knowledge vs. management knowledge* [Grunstein and Barthès, 1996]: the technical knowledge is used by the core business in its day-to-day work. The strategic or management knowledge is used by the managers to analyse and the organisation functioning and build management strategies

And so on: content *vs.* context [Dzbor *et al.*, 2000], explicable knowledge (with justifications) *vs.* not explicable (bare facts), granularity (fuzzy *vs.* precise, shallow *vs.* deep, etc.), individual *vs.* collective, volatile *vs.* perennial (from the source point of view), ephemeral *vs.* stable (from the content point of view), specialised *vs.* common knowledge, public *vs.* personal/private, etc.

A first remark is that the differences proposed here are not always mutually exclusive and neither always compatible. Secondly a knowledge domain is spread over a continuum between the extremes proposed here. However, most traditional company policies and controls focus on the tangible assets of the company and leave unmanaged their important knowledge assets [Macintosh, 1994].

An organisational memory may include (re)sources at different levels of the data-information-knowledge scale and of different nature. It implies that a management solution must be able to handle and integrate this heterogeneity.

1.2.2 Knowledge management and organisational memory lifecycle

The stake in building a corporate memory management system is the coherent integration of this dispersed knowledge in a corporation with the objective to "promote knowledge growth, promote knowledge communication and in general preserve knowledge within an organisation" [Steels, 1993]. This implies a number of activities to turn the memory into a living object.

In [Dieng *et al.*, 2001] my colleagues discussed the lifecycle of a corporate memory. I added the overall activity of managing the different knowledge management processes *i.e.,* as described in [Zacklad and Grundstein, 2001a; 2001b] to promote, organise, plan, motivate, etc. the whole cycle. The result is depicted in Figure 2 and I shall comment the different phases with references to the literature.

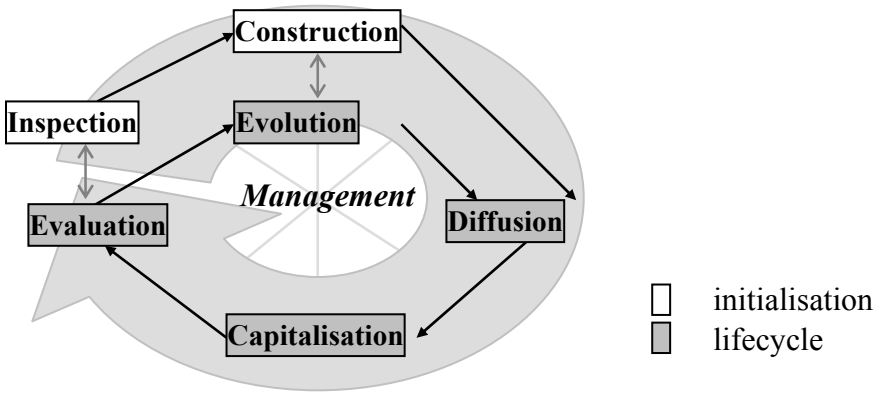


Figure 2 Lifecycle of a corporate memory

Here are some cycles proposed in literature:
- [Grunstein and Barthès, 1996]: locate crucial knowledge, formalise / save, distribute and maintain, plus manage that was added in [Zacklad and Grundstein, 2001a; 2001b]
- [Zacklad and Grundstein, 2001a; 2001b]: manage, identify, preserve, use, and maintain.
- [Abecker *et al.*, 1998]: identification, acquisition, development, dissemination, use, and preservation.
- [Pomian, 1996]: identify, acquire/collect, make usable.
- [Jasper *et al.*, 1999]: create tacit knowledge, discover tacit or explicit knowledge, capture tacit knowledge to make it explicit, organise, maintain, disseminate through push solutions, allow search through pull solutions, assist reformulation, internalise, apply
- [Dieng *et al.*, 2001]: detection of needs, construction, diffusion, use, evaluation, maintenance and evolution.

I have tried to conciliate them in the schema in Figure 2, where the white boxes are the initialisation phase and the grey ones are the cycle strictly speaking. The central 'manage' activity oversees the others. I shall describe them with some references to the literature:
- *Management*: knowledge management is difficult and costly. It requires a careful assessment of what knowledge should be considered and how to conduct the process of capitalising such knowledge [Grunstein and Barthès, 1996]. All the following activities have to be planned and supervised; this is the role of a knowledge manager.
- *Inspection*: inventory of fixtures to identify knowledge that already exists and knowledge that is missing [Nonaka, 1991]; identify strategic knowledge to be capitalised [Grundstein and Barthès, 1996]; make an inventory and a cartography of available knowledge: identify assets and their availability to plan their exploitation and detect lacks and needs to offset weak points [Dieng *et al*., 2001]

- *Construction*: build the corporate memory and develop necessary new knowledge [Nonaka, 1991], memorise, link, index, integrate different and/or heterogeneous sources of knowledge to avoid loss of knowledge [Dieng *et al*., 2001]
- *Diffusion*: irrigate the organisation with knowledge and allocate new knowledge [Nonaka, 1991]. Knowledge must be actively distributed to those who can make use of it. The turn-around speed of knowledge is increasingly crucial for the competitiveness of companies. [Van Heijst *et al.*, 1996] Make the knowledge flow and circulate to improve communication in the organisation: transfer of the pieces of knowledge from where they were created, captured or stored to where they may be useful. It is called "activation of the memory" to avoid oblivion knowledge buried and dormant in a long forgotten report [Dieng *et al.*, 2001]. This implies a facility for deciding who should be informed about a particular new piece of knowledge and this point justify this sections about user modelling and organisation modelling.
- *Capitalisation*: process which allows to reuse, in a relevant way, the knowledge of a given domain, previously stored and modelled, in order to perform new tasks [Simon, 1996], to apply knowledge [Nonaka, 1991], to build upon past experience to avoid reinventing the wheel, to generalise solutions, combine and/or adapt them to go further and invent new ones, improve training and integration of new members [Dieng *et al.*, 2001] The use is tightly linked to diffusion since the way knowledge is made available conditions the way it may be exploited.
- *Evaluation*: it is close to inspection since it assesses the availability and needs of knowledge. However it also aims at evaluating the solution chosen for the memory and its adequacy, comparing the results to the requirements, the functionalities to the specifications etc.
- *Evolution*: it is close to construction phase since it will deal with additions to the current memory. More generally it is the process of updating changing knowledge and removing obsolete knowledge [Nonaka, 1991]; it is where the learning spiral takes place to enrich/update existing knowledge (improve it, augment it, precise it, re-evaluate it etc.).

The memory management consists of two initial activities (inspection of the knowledge situation and construction of the initial memory) and four cyclic activities (diffusion, capitalisation, evaluation, evolution); all these activities being planned and supervised by the management.

In his third axiom, [Euzenat, 1996] insisted on the collaborative dimension: people must be supported in discussing about the knowledge introduced in the knowledge base. In this perspective, re-using, diffusing and maintaining knowledge should be participatory activities and underlines the strong link between some problems addressed in knowledge management and results of research in the field of Computer Supported Collaborative Work. "Users will use the knowledge only if they understand it and they are assured that it is coherent. The point is to enforce discussion and consensus while the actors are still at hand rather than hurrying the storage of raw data and discovering far latter that it is of no help." [Euzenat, 1996]

[Nonaka and Takeuchi, 1995] details the social and individual processes at play during the learning spiral of an organisation and focuses on the notion of tacit and explicit knowledge (Figure 3).

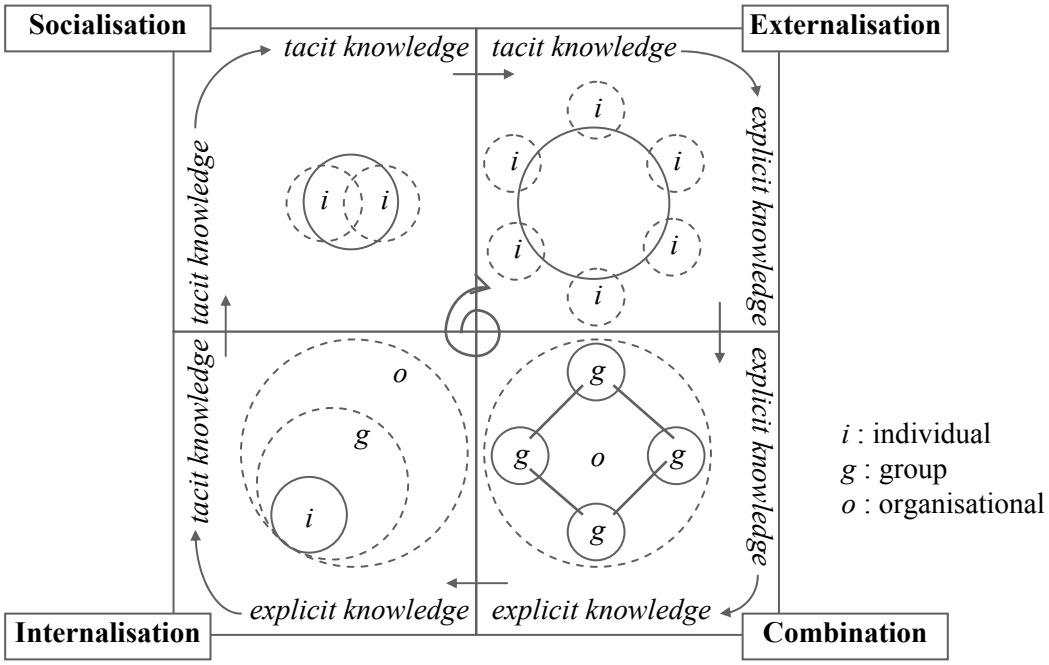


Figure 3 Learning spiral of [Nonaka and Konno, 1998]

[Nonaka and Takeuchi, 1995] believe that knowledge creation processes are more effective when they spiral through the four following activities to improve the understanding:

- *Socialisation* (Tacit → Tacit): transfers tacit knowledge in one person to tacit knowledge in another person. It is an experiential and active process where the knowledge is captured by walking around, through direct interaction, by observing behaviour by others and by copying their behaviours and beliefs. It is close to the remark of [Kimble *et al.*, 2001] saying that communities of practice are central to the maintenance of soft knowledge.
- *Externalisation* (Tacit → Explicit): getting tacit knowledge into an explicit form so that you can look at it, manipulate it and communicate it. It can be an individual articulation of one's own tacit knowledge where one is looking for awareness and expressions of one's ideas, images, mental models, metaphors, analogies, etc. It can also consist in eliciting and expressing the tacit knowledge of others into explicit knowledge.
- *Combination* (Explicit → Explicit): take explicit explainable knowledge, combine it with other explicit knowledge and develop new explicit knowledge. This is where information technology is most helpful, because explicit knowledge is conveyed in artefacts (e.g. documents) that can be collected, manipulated, and disseminated allowing knowledge transfer across organisations.
- *Internalisation* (Explicit → Tacit): understanding and absorbing collectively shared explicit knowledge into individual tacit knowledge actionable by the owner. Once one deeply learned a process, it becomes completely internal and one can apply it without noticing, as reflex or automatic natural activities. Internalisation is largely experiential through the actual doing, in real

situation or in a simulation. A symptom is that generally when one tries to pay attention to how one does these things, it impairs one's performance and this is a break to elicitation processes.

Finally, concerning the learning, [Van Heijst *et al.*, 1996] differentiate two types:
- top-down learning, or strategic learning: at some management level a particular knowledge area is recognised as strategic and deliberate action is planed and undertaken to acquire that knowledge.
- bottom-up learning: a worker learns something which might be useful and that this lesson learned is distributed through the organisation.

1.2.3 Typology or facets of organisational memories

As far as I know and so far, no one has tackled all the different types of knowledge and activities of management in one project coming up with a complete solution. Every study and project focused on selected types of knowledge and knowledge management activities. As a result we can find a typology of memories and management systems.

From an external point of view, [Barthès, 1996] gives several facets of the corporate memory problem that can be considered: socio-organisational, economic, financial, technical, human and legal.

From on internal point of view, the first seminal typology is the one of [Van Heijst *et al.*, 1996]. It is based on the envisaged management processes as shown in (Table 1).

	Passive collection	**Active collection**
Passive distribution	Knowledge attic	Knowledge sponge
Active distribution	Knowledge publisher	Knowledge pump

Table 1 Types of organisational memories [Van Heijst *et al.*, 1996]

The *knowledge attic* is a corporate memory used as an archive which can be consulted when needed and updated when wanted. It is not intrusive, but it requires a high discipline of the organisation members not to become obsolete.

The *knowledge sponge* is a corporate memory actively fed to keep it more or less complete. Its use is left to the individual responsibility of organisation members.

The *knowledge publisher* is a corporate memory where the contribution is left to the individual workers while memory maintainers analyse the incoming knowledge and combine it with the stored one and forward the relevant news to potentially interested members.

The *knowledge pump* is a corporate memory that ensures that the knowledge developed in the organisation is effectively captured from and used by members of the organisation.

Other researchers classify memories depending on the knowledge content:
- *trade/profession/ technical memory*: composed of the referential, the documents, the tools and methods used in a given profession [Tourtier, 1995]. This knowledge about a domain, the investigations and research results [Pomian, 1996] the knowledge used everyday inside the organisation by its members to perform their daily activity [Grundstein and Barthès, 1996].
- *managerial memory*: related to organisation, activities, products, participants [Tourtier, 1995], about the organisation itself, its structure, its arrangement, its management principles, its policy, its history, [Pomian, 1996]. It captures the past and present organisational structures of the enterprise (human resources, management, etc [Grundstein and Barthès, 1996]. This memory is extremely close to the field of organisation modelling I shall describe in a following section.
- *individual memory*: characterised by status, competencies, know-how, activities of a given member of the enterprise, [Tourtier, 1995].

- *project memory*: it is acquired in the context of a project that must be saved with the knowledge to preserve its meaning [Pomian, 1996]. It comprises the project definition, activities, history and results [Tourtier, 1995] It is used to capitalise lessons and experience from a given projects because although the ephemeral nature of the project is seductive from an organisational point-of-view to augment flexibility, adaptability, etc. On the other hand the memory of the project suffers from volatility. The project memory preserve the technical memory and managerial memory mobilised for a project. It can be the memory of an on-going project or a past project. In any case, it is important to capture the course/progress, rationale and the context [Pomian, 1996].

Finally, one can classify the memories on their form and additional characteristics:

- *Non computational memory*: A non computational memory [Dieng *et al.* 1999] is made of physical artefacts (paper-based documents, video tapes, etc.) capturing knowledge that had never been elicited previously. [Dieng *et al.* 1999] distinguish two different aims to build such a memory: to elaborate synthesis documents on knowledge that is not explicit in reports or technical documentation, and is more related to the know-how of the experts of the enterprise; to improve enterprise production through experts' propositions on their tasks in a design process.
- *Data warehouses* and *Data marts*: In many companies, one of the first KM tools is a data warehouse, *i.e.,* a central storage area for an organisation's transaction data [O'Leary, 1998]. It usually replicates or at least accesses content from many of the organisation's databases. From this, it provides with wide variety of data and it tries to present a coherent picture of business conditions at a single point in time. Automatic report generation systems based on queries and views of the databases as well as more complex knowledge discovery and data mining techniques may be used to enable knowledge workers (essentially managers) to collect information supporting management decision-making. *Data marts* are like data warehouse, but usually smaller, they focus on a particular subject or department; they may be subsets of larger data warehouses. They are usually linked to a community of knowledge workers.
- *Internal vs. external memories*: a corporate memory may not be restricted to the sole organisation [Rabarijaona *et al.*, 1999]. An internal corporate memory can rely on an internal resources while an external corporate memory rather includes information and knowledge stemming from the external world, but useful for the organisation's activities. The retrieval and integration of information available on the Web may be interesting and I call it a corporate portal to the outside "world *wild* web" *i.e.,* a system that uses the organisation's memory as a filter to tame the heterogeneity and information overload of the Web.
- *Document-based memory or knowledge warehouses*: it relies on existing documents to build a memory. The construction of such a memory begins by the collection of the different documents and requires an interface to manage them (addition of documents, retrieval of documents, etc.) [Dieng *et al.* 1999]. "A good documentation system is very likely the least expensive and the most feasible solution to knowledge management" [Poitou, 1995]. This notion is close to the knowledge warehouses of [O'Leary, 1998] that are aimed at qualitative data and contains manuals and design rules, specifications, and requirements. [O'Leary, 1998] explains that historically, Lotus Notes was one of the primary tools for storing qualitative and document-based information and for facilitating virtual groups. Now low cost Web-based solutions within intranet environments have become the favourite solution. [Rabarijaona *et al*, 1999] identified several kinds of documents that can be exploited in a document-based corporate memory:
  - documents linked to projects: specifications of the product to be designed or manufactured, design documents, test documents, contractual technical reports,
  - reference bibles in a given profession,
  - visual documents such as photos, scanned plans, iconographic documents,
  - technical reports, scientific or technical articles,
  - books, theses, norms, archive documents, guides, dossiers of technological intelligence,
  - on-line documentation, user manuals, reference manuals, business dossiers, etc.
- *Knowledge-based Corporate Memory*: such a memory is based on elicitation and explicit modelling of knowledge from experts [Dieng *et al.* 1999]. It can be mixed with the previous document-based memory by indexing documents through a formal representation of knowledge underlying them.

However, the goal of this approach is to provide assistance to users, supplying them with relevant corporate information, but leaving them the responsibility of a contextual interpretation and evaluation of this information [Kuhn and Abecker, 1997].

- *Case-based memories*: organisations have a collection of past experiences (successes or failures) that can be represented explicitly in a same formalism to compare them [Dieng *et al.* 1999]; these formalised experiences are called cases and their management can exploit case-based reasoning [Simon and Grandbastien, 1995; Simon, 1996]. [Dieng *et al.* 1999] distinguish two aims: avoid the scattering of the expertise by concentrating knowledge of all experts in dedicated cases; allow a continuous evolution of the memory thanks to the progressive addition of new cases. Case-based reasoning allows to capitalise upon cases already encountered, in order to solve new ones. The retrieval mechanism is built around a similarity measure to find past cases close enough to suggest a solution that could be reused or adapted to a new problem to be solved. This approach is very useful for project memories *project memories*.
- *Distributed Memory*: a distributed memory is interesting for supporting collaboration and knowledge sharing between people or organisations dispersed geographically. It is essential for virtual enterprises made of distributed organisations and teams of people that meet and work together online [Dieng *et al.*, 1999]. Distributed memory naturally relies on internet and Web technologies.
- *People-based memory*: individuals represent a prime location where intellectual resources of an organisation are located [Dzbor *et al.*, 2000]. Thus another frequently used corporate application is a human resource knowledge base about employee capabilities and skills, education, specialities, previous experience, etc. [O'Leary, 1998] Although it is important to disembody and materialise the knowledge to make it perennial and perpetuate it, it is also clear that, so far, not all the knowledge can be captured in a symbolic system to be memorised. In that case it is important to capture the identity of the sources (e.g. an expert) and index this system-external sources to include it in the overall memory and also to know which part of the current memory are not disembodied and safely stored. [Drucker, 1994] states how if knowledge is the key concept in our future society, the 'person' will be central because knowledge, although stored in books and databases, is actually embodied in a person. [Liao *et al.*, 1999] proposed a competence knowledge base system for the organisational memory to facilitate finding of appropriate contact person. Another type of expert matchmaker system is proposed by [Sangüesa Sol and Pujol Serra, 1999]: it builds a representation of acquaintance networks by mining Web page references. Other systems uses document co-authoring, recommending systems, etc.

Although this last categorisation is quite technical, one should not reduce knowledge management to a technical problem. Traditionally, enterprises have addressed knowledge management from either a management or a technological point of view. [Abecker *et al.*, 1998]. The set-up of an organisational memory requires an approach balanced and integrated in its 'multidisciplinarity' [Dieng *et al.*, 2001].

Moreover, according to [Ackerman and Halverson, 1998] there does not exist *an* organisational memory, but rather a supra-individual memory, using several people and many artefacts; this vision is close to the one adopted in distributed cognition. These authors believe that this view of a network of artefacts and people, of memory and of processing, bound by social arrangements, provides a deeper and ultimately more usable understanding of organisational life. Effective management of knowledge requires hybrid solutions that involve both people and technology [Davenport, 1996]. The role of humans in this partnership is to solve ill-defined problems and derive innovative and creative solutions using knowledge preserved and provided by the memory. Such an idea supersedes traditional expert-systems that were acting more autonomously, but were lacking many essential human capabilities. Organisational memories are typical example of an intelligent assisting computer- (network-) based tool [Dzbor *et al.*, 2000].

The different types or aspects I compartmentalised here are not mutually exclusive. Hybrid systems are being studied and developed with different dimensions and interesting problems of interactions between these dimensions arise, e.g: [Nagendra Prasad and Plaza, 1996] studied corporate memories as distributed case bases.

A memory is likely to be an integrated hybrid system both from the technical (several technical approaches may be combined) and physical (documents, software and other artefacts, people, communities, etc. are involved in a solution) points of view. It requires an approach balanced and integrated in its 'multidisciplinarity'.

1.3 Organisation modelling

I do not intend to make a complete state of the art of enterprise modelling, because this field is very large and most of the contributions are noticeably far from my concerns. However some interesting points have been selected and are reported here since they are related to the notion of Organisational Memories.

1.3.1 Origins of organisational model

For [Rolstadås, 2000], a model is an abstract and limited representation of a piece of reality expressed in terms of some formalism and that can be used to obtain information about that piece of reality. Therefore, an organisation model is used to give a limited representation of an organisation. [Rolstadås, 2000] quotes several definitions of the enterprise model: some tend to adopt more generic definitions than others, they vary in their focus and in their definition of an enterprise: some have single view while others handle multiple views, and so on.

[Solberg, 2000] explains with reference to [Vernadat, 1996] that organisation modelling is the set of activities, methods and tools related to developing models for various aspects of an organisation. He believes that such a model exists in any organisation, be it small or large, but that it is poorly formalised: organisation charts, documented operational procedures, regulation texts, databases, knowledge bases, data files, application programs and to a large extend in the minds of members of the organisation. "Methods and tools are required to capture, formalise, maintain, and use this knowledge for better operation and control of complex systems such as manufacturing enterprises." [Solberg, 2000]

For [Szegheo, 2000] an organisation model can be made of several sub-models including (but not limited to) process models, data models, resource models and structural models. The organisation can be viewed from different aspects and the author underlines that in practice all these aspects cannot be taken into account in one model since the result would be too complex to handle and work with. The objective of a model is "neither to fully describe all aspects of a manufacturing enterprise nor to model the entire enterprise. This would be useless, nearly impossible, and certainly endless as enterprises are tremendously complex systems in terms of number of entities involved, things to do, decision variables to be considered, and processes to be controlled." [Solberg, 2000]

Usually the model contains those aspects that are crucial for solving the problem that is being considered [Szegheo, 2000] *i.e.,* the model depends on the task it is used for. "The degree of abstraction and simplification depends on the interest of the targeted audience" [Szegheo, 2000]. Thus the model depends on the stakeholders of the application scenario it was designed for.

To generalise, the degree of abstraction simplification, as well as the points of view adopted, depends on the specifications of the (computerised or not) system exploiting the formal model, and therefore, it ultimately depends on the stakeholders' expectation. Therefore, [Solberg, 2000], with reference to [Vernadat, 1996], insists on the fact an enterprise model must have a finality defined by the goal of the modeller. He gives examples of such finalities:
- to better represent and understand how the organisation or some part(s) of it works,
- to capitalise acquired knowledge and know-how for later reuse,
- to rationalise and secure information flows,
- to design or redesign and specify a part of the organisation,
- to analyse some aspects of the organisation,
- to simulate the behaviour of some part(s) of the organisation,
- to make better decisions about organisation operations and organisation,
- to control, co-ordinate, or monitor some parts of the organisation.

Organisation modelling is currently extensively used for organisation design concerns. Examples of problems addressed by techniques using organisation modelling are:
- organisation development (e.g. [Alfines, 2000])
- organisation integration (e.g. [Røstad, 2000])
- organisation simulation (e.g. [Szegheo and Martinsen, 2000])
- performance measurement (e.g. [Deng, 2000])
- self-assessment (e.g. [Fagerhaug, 2000])
- business process improvement (e.g. [Andersen, 2000])
- setting-up extended organisation (e.g. [Szegheo and Petersen, 2000])

It is clear that any kind of organisation model serves a purpose. There are many different purposes, but fundamentally any model aims to "make people understand, communicate, develop, and cultivate solutions to business problems [the difference between different models] might lay in the purpose of the model, the content of the model, the quality of the formalism and manifestation, the level of abstraction, and the span of existence." [Szegheo, 2000]

Thus, so far, the enterprise modelling field has been mainly concerned with simulation and optimisation of the production system design with relevant criteria called performance indicators. Such modelling aims to improve industrial competitiveness. It provides benchmark for business processes and is used for business process re-engineering.

As it is largely acknowledged in contemporary literature, 'globalisation' and 'information society' modified the market rules of the game and set new constraints on its stakeholders. [Rolstadås, 2000] notices "there is an industrial change in direction of organising work in projects. This change from operations management to project management involves that enterprises to a larger extent will handle their business as projects and use project planning and control tools rather than the classic operations management tools". In fact, this introduces the necessity of being able to create, manage and dissolve ephemeral teams when necessary to adapt the dynamic of market. One of the new stakes of this situation is to be able to capitalise and reuse the knowledge from past project experiences when their structure (team) has dissolved in a new organisation.

[Rolstadås, 2000] also identifies "a trend toward organising work to use teams that are designed on an interdisciplinary basis. This enables things to be done more in parallel than earlier and thus reduces time to market. It also stimulates new innovation, often in the intersection between technology and social sciences". This new trend leads to the problem of managing and integrating multiple expertise points of view in the design rationale and then in the corporate memory to enable the history of an older project to be revisited and to take advantage of this experience in new projects. "The virtual nature of the agile organisation entails a greater degree of communication, co-ordination, and co-operation within and among enterprises; the agile organisation must be integrated from the structural, behavioural, and informational point of view. (...) The drive for more agile enterprises requires a degree of integration that is not possible with-out the use of a sophisticated information infrastructure. At the core of this infrastructure lies an enterprise model." [Fox and Gruninger, 1998]

Another trend is the lifecycle follow-up aspect. "This takes environment and sustainability into account. Products must be made for the entire lifecycle including scrapping, disassembly, or recycling." [Rolstadås, 2000]. The lifecycle aspect also implies the transfer of information or knowledge from one stage to another (e.g.: from assembly to disassembly) and therefore, it sets constraints on the documentation and more broadly the memory attached to one product.

"An enterprise model is a computational representation of the structure, activities, processes, information, resources, people, behaviour, goals, and constraints of a business, government, or other enterprise. It can be both descriptive and definitional - spanning what is and what should be. The role of an enterprise model is to achieve model-driven enterprise design, analysis, and operation." [Fox and Gruninger, 1998] Now organisations are aware that their operation is strongly influence by the knowledge management. Therefore, the organisation model has a role to play in knowledge management solutions too. The new trends exposed by Rolstadås and the shift in the market rules led

enterprises to become aware of the value of their memory and the fact that enterprise model has a role to play in this application too. [Rolstadås00] notices that enterprise models may well constitute a theoretical basis for the information system in an enterprise, and are regarded by many as a substantial opportunity to improve global competitiveness of industry. Achieving organisational integration requires an "infrastructure that supports the communication of information and knowledge, the making of decisions, and the co-ordination of actions. At the heart of this infrastructure lies a model of the enterprise. (...) It would not be overly general to say that most information systems in use within an enterprise incorporate a model of some aspect of the enterprise's structure, operations, or knowledge." [Fox and Gruninger, 1998] However, this modelling needs to be systematic, explicit and integrated to the whole information organisational system.

1.3.2 Ontology based approaches for organisation modelling

As noticed by [Szegheo, 2000] the enterprise model, like any model, will have to be expressed in terms of a language. The language could be formal or informal. The richest languages are natural languages, their use would seem logical. The problem is that they lack formalism and their interpretation is not universal. A good modelling language is formalised and its usage and meaning are unambiguous. As stressed later ontologies are used to capture the intended and unambiguous meaning of the modelling primitives. An organisational ontology defines the concepts relevant for the description of an organisation, e.g.: organisational structure, processes, strategies, resources, goals, constraints and environment. Such an ontology can be used to make explicit models for automating business-engineering or supporting the exchange of information and knowledge in the enterprise [Fraser, 1994]. Moreover, this representation should eliminate much of the programming required to answer "simple" common sense questions about the enterprise [Fox and Gruninger, 1998] by capturing the essential characteristics of the entities existing in an organisation and their relations.

An organisational model is an explicit representation of the structure, activities, processes, flows, resources, people, behaviour, goals, and constraints of an organisation. The corresponding ontology captures the essential characteristics of the modelled entities and forms of relations existing between them in an unambiguous consensual vocabulary.

Here is an overview of some approaches that use a more or less explicit ontology. It is essentially based on the excellent article of [Fox and Gruninger, 1998] where a more complete version can be found:

- *CIMOSA* (Computer-Integrated Manufacturing—Open-System Architecture) [AMICE, 1993] [Bernus *et al.*, 1996]: integration of enterprise operations by means of information exchange within the enterprise. It defines an integrated methodology to support all phases of a CIM system lifecycle from requirements specification through system design, implementation, operation, and maintenance. This description is used to control the enterprise operation and to plan, design, and optimise updates of the real operation environment. It defines four different views concerned with the enterprise: (1) functional structure, the related control structure and the workflow, (2) information required by each function, (3) the resources and their relationship to other structures (4) enterprise organisational structures and responsibilities.
- *Enterprise Ontology* [Uschold *et al.*, 1998]: this ontology supports integrating methods and tools for capturing and analysing key aspects of an enterprise. This ontology is semiformal; it provides a glossary of terms expressed in a restricted and structured form of natural language supplemented with a few formal axioms. It comprises of five parts: (1) metaontology: entity, relation-ship, role, actor, and state of affairs; (2) activities and processes: activity, resource, plan, and capability; (3) organisation: organisational unit, legal entity, management, and owner-ship; (4) strategy: purpose, strategy, help to achieve, and assumption; (5) marketing: sale, product, vendor, customer, and market.

- *Enterprise-wide Data Modelling* [Scheer, 1989]: this ontology undertakes to construct data structures for typical functional areas (departments), such as production, engineering, purchasing, human resources, sales and marketing, accountancy, and office administration, with the aim of supporting planning, analysis, and traditional accounting systems in general. It uses the entity-relationship model to systematically develop the data structures for the enterprise in terms of entities (something that can be identified in the users' work environment), attributes (characteristics-properties of an entity), and relationships (the association of entities).
- *GERAM* (Generic Enterprise Reference Architecture And Methodology): this ontology is about integrated enterprise [Bernus, *et al.*, 1996]. Its coverage is: products, enterprises, enterprise integration, and strategic enterprise management.
- *IDEF* Ontology: the ontology developed at KBSI are intended to provide a rigorous foundation for the reuse and integration of enterprise models [Fillion *et al.*, 1995]. The ontology is a first-order theory consisting of a set of foundational theories, along with a set of ENTERPRISE models.
- *MILOS* [Maurer and Dellen, 1998]: this approach relies on a process-oriented view on the organisation inspired of research on workflow management: for example, it offers a process modelling language for representing knowledge upon work processes (e.g. "process, product and resource models, project plans and schedules, products developed within projects, project traces, background knowledge such as guidelines, business rules, studies").
- *NIST Process-Specification Language* [Schlenoff *et al.*, 1996]: to facilitate complete and correct exchange of process information among manufacturing applications (e.g. scheduling, process planning, simulation, project management, work flow, business-process reengineering, etc.). The core provides the basis for representing the simplest processes (e.g. time, resource, activity). The outer is for describing common processes (e.g. temporal constraints, resource grouping, alternative tasks). Extensions group representation primitives for particular sets of applications that provide added functions (e.g. goals, intentions, organisation constraints, products). Application-specific correspond to a specific application.
- *OLYMPIOS* [Beauchène *et al.*, 1996]: this approach models an enterprise organisation, using a model stemming from quality management and focusing on «customer-supplier» relationships between the enterprise members.
- *PERA* (Purdue Reference Architecture): this approach is interested in enterprise modelling for a computer-integrated manufacturing (CIM) [Bernus *et al.*, 1996] [Williams, 1991]. The functional descriptions of the tasks and functions of the enterprise are divided into two major streams: (1) decision, control, and information and (2) manufacturing, and customer service.
- *Process-Interchange Format* (PIF): it is an interchange format to support the automatic exchange of process descriptions among a wide variety of business-process modelling and support system (e.g. workflow tools, process simulation systems, business-process reengineering tools, process repositories). PIF serves as a common format to support inter-operability. PIF is a formal ontology structured as a core plus a set of extensions. The top-level defines classes "activity, object, agent, and time point" as well as the relations "performs, uses, creates, modifies, before, and successor ".
- *TOVE* (Toronto Virtual Enterprise Ontology): the goal of TOVE [Fox *et al.*, 1993] is to create an ontology for both commercial and public enterprises that every application can jointly understand and use with the meaning of each term precise and unambiguous. The associated test bed provides an environment for analysing enterprise ontologies; it provides a model of an enterprise and tools for browsing, visualisation, simulation, and deductive queries. TOVE currently spans knowledge of activity, time, and causality, resources, cost, quality, organisation structure, product, and agility. TOVE aims at creating a generic, reusable enterprise data model that has the following characteristics: it provides a shared terminology for the enterprise that each agent can jointly understand and use; it defines the meaning of each term (aka semantics) in a precise and as unambiguous manner as possible; it implements the semantics in a set of axioms that will enable to automatically deduce the answer to many "common sense" questions about the enterprise; it defines a symbology for depicting a term or the concept constructed thereof in a graphical context. The TOVE reusable representation is a significant ontological engineering of industrial concepts.

1.3.3 Concluding remarks

My first remark is on the paradoxical aspect of modelling which arises as soon as a model has to be used by people that may not have been involved in its design or when the design is subject to a consensus. [Solberg, 2000], again with reference to [Vernadat, 1996], explained it perfectly:

> "Enterprise models are useful only if they are used. They will be accepted by users as a tool if they are simple to understand, easy to use, computer supported, and if they provide a realistic image of the reality. This explains the failure of many approaches proposed in the past, or the difficulty of proven sophisticated techniques to be accepted in practice, such as Petri nets.
>
> The opposite side of the coin is that users are often looking for oversimplified techniques, which do not go far enough in details and at the end have little value. The difficulty for tool builders is to develop sophisticated modelling and analysis environments which hide this complexity and have a user-friendly interface, good graphical model representations, and 'talk' the language of the user while at the same time offering powerful analysis and simulation capabilities.
>
> Ultimately, the success of an enterprise model depends on if it works appropriately, and the best way to find this out is to test it. Such a test will uncover how the enterprise model works." [Solberg, 2000]

My second remark is that in this Ph.D., my goal was not to evaluate the model of an organisation or optimise it to support enterprise evolution. The model I envisaged aimed at supporting corporate memory activities involved in the application scenario (this is why the enterprise modelling state of the art is focused and limited). As [Papazoglou and Heuvel, 1999] stressed, it is necessary for information systems to have an understanding of the organisational environment, its goals and policies, so that the resulting systems will work effectively together with human agents to achieve a common objective. Enterprise modelling will enable the system to get insight in its environment (organisational structure,...); to give the system an awareness of the organisation it is inhabiting.

1.4 User-modelling and human-computer interaction

User modelling started in the early 80's and human-computer interaction in the 60's. It would be a tremendous work and a task completely out of the scope of this Ph.D. to make a comprehensive state of the art of these two domains. However, in a computer assisted solution of corporate memory management, the interaction with users has to be studied. The two fields are extremely linked both historically and in their research objectives; in human-computer interaction, the human is quite often a user whose model must be taken into account to improve the behaviour of the system. In a knowledge management perspective, the user is part of the context, and the context is an important factor when knowledge is handled, therefore, user modelling has a role to play in knowledge management solutions. On the other side, the problem of modelling humans and their cognitive activities will raise considerations that fall within the competence of knowledge representation and knowledge-based systems. This the two domains can complement each other.

1.4.1 Nature of the model

The first aspect of the problem is the nature and content of the model. There are two types of models: individual models and group or stereotype models.

An *individual user model* consists of representation of assumptions about one or more types of user characteristics in models of individual users [Kobsa, 2001]. It includes assumptions about their knowledge, beliefs, goals, preferences, interests, misconceptions, plans, tasks, and abilities, the work context, etc. The forms that a user model may take are as varied as the purposes for which user models are formed. User models may seek to describe: the cognitive processes that underly the user's actions; the differences between the user's skills and expert skills; the user's behavioural patterns or preferences; or the user's characteristics. [Webb *et al.*, 2001] With reference to [Kobsa *et al.*, 1999], Brusilovsky [2001] suggests to distinguish adaptation to user data, usage data, and environment data. User data comprise the adaptation target, various characteristics of the users. Usage data comprise data about user interaction with the systems that cannot be resolved to user characteristics. Environment data comprise all aspects of the user environment that are not related to the users themselves (e.g. user location and the user platform). [Brusilovsky, 2001] divides the user data into:
- *User Characteristics* (user's goals/tasks, knowledge, background, experience, and preferences),
- *User interests* (long-term interests such as a passion and short-term interest such as search goal),
- *User's individual traits* (e.g. personality factors, cognitive factors, and learning styles.).
- *User groups and stereotypes* model are the representation of relevant common characteristics of users pertaining to specific user subgroups of the application system [Kobsa, 2001].

A user model is the explicit representation of assumptions or facts about a real user or a stereotype. It includes the user's characteristics (assumptions about the knowledge, beliefs, goals, preferences, interests, plans, tasks, and abilities, work context, etc.), the past usage, and the environment. *NB: A user model may be part of an organisational model.*

1.4.2 Adaptation and customisation

Based on models, there exist two categories of adaptation of human-computer interaction:
- *Manual customisation*: offer the users the capability to select, or set between different alternative interaction characteristics, among the ones built into the system. [Stephanidis, 2001]
- *Automatic adaptation*: the system is capable of identifying those circumstances that require adaptation, and accordingly, select and effect an appropriate course of action. This implies that the system possesses the capability to monitor user interaction, and use the monitoring data as the basis upon which it draws assumptions, continuously verifying, refining, revising, and, if necessary, withdrawing them. [Stephanidis, 2001]

Of course the latter is the most complex and the richest one and it is on this point that the field focuses. It can be applied both at the individual or collective level.

[Zukerman and Albrecht, 2001] differentiates between the content-based approach where the behaviour of users is predicted from their past behaviour, and the collaborative approach where the behaviour of users is predicted from the behaviour of other like-minded people:
- *Content-based learning* is used when users' past behaviour is a reliable indicator of their future behaviour so that a predictive model can be built. This approach is ideal for tailoring a system's behaviour to the specific requirements of a particular user, but it requires each user to provide relatively large amounts of data to enable the construction of the model. In addition, the selected features have a substantial effect on the usefulness of the resulting model. If they are too specific, the system is useful only for repetitive behaviours, while if they are too general, predictions are of debatable usefulness [Zukerman and Albrecht, 2001].
- *Collaborative learning* is used when a user behaves in a similar way to other users. A model is built using data from a group of users, and it is then used to make predictions about a particular individual user. This approach reduces the data collection burden for individual users, and can be implemented using the specific values of the data without obtaining features with the "right" level of abstraction. However, it does not support tailoring a system to the requirements of a particular user and there is a risk to conflate all users into a model that represents an "average user" [Zukerman and Albrecht, 2001].

[Fischer, 2001] also distinguishes between three techniques to find out what the user really knows and does: being told by the users (e.g. by questionnaires, setting preferences, or specification components; being able to infer it from the user's actions (e.g. by using critics) or usage data; and communicating information about external events to the system. However, as noticed by Zukerman and Albrecht, these approaches have complementary advantages and disadvantages that call for a solution combining both types of modelling approaches.

A number of goals can be pursued exploiting these models such as:
- *user classification*: classification of users as belonging to one or more subgroups, and the integration of the typical characteristics of these subgroups into the current individual user model [Kobsa, 2001].
- *collaborative or clique-based processing* by comparison of different users' selective actions: users' selective actions are matched with those of other users, and users' future selective actions are predicted based on those of the most similar other users [Kobsa, 2001].
- *user prediction or simulation*: formation of assumptions about the user, based on the interaction history in order to predict future action or simulate the user's behaviour.

In any goal, a recurrent problem to address is the one of *detecting patterns in user models and behaviour*. To that purpose, user-modelling techniques draw upon machine learning, probabilistics, and logic-based methods. Several different statistical models have been used in the framework of both the content-based and the collaborative approach. The main models are: linear models, TF*IDF-based models, Markov models, neural networks, classification and rule-induction methods,

Dempster-Shafer theory and Bayesian networks [Zukerman and Albrecht, 2001]. Using machine learning, user's behaviour provide training examples that are then used to form a model designed to predict future actions.

However several problems challenge the domain, namely: the need for large data sets; the need for labelled data; concept drift *i.e.,* the changing nature of the knowledge learned; and the computational complexity [Webb *et al.*, 2001]. Moreover, whereas much of the academic research in machine learning for user-modelling concentrates on modelling individual users, many of the emerging applications in electronic commerce relate to forming generic models of user communities [Webb *et al.*, 2001].

Plan recognition is a special kind of pattern detection which tries to recognise the users' goals and their intended plan for achieving them, the motto here is to try to build "systems that do what you mean, not what you say". Most plan recognition systems start with a set of goals that a user might be expected to pursue in the domain and with an observed action by the user. Then, the plan recognition system infers the user's goal and determines how the observed action contributes to that goal. For this, it needs a set of actions that the user might execute in the domain and a set of recipes that encode how a user might go about performing these actions. Recipes constitute a plan library and include for each action, the preconditions, the sub-goals, and the effects of executing the action. Classical reasoning uses chaining algorithms between preconditions and post-conditions and a wide variety of mechanisms have been proposed for narrowing the space of viable hypotheses about a user's plan. The reasoning performed by the system uses domain-dependent knowledge, but it is itself largely domain-independent [Carberry, 2001].

Here again hybrid solutions are being envisaged. In particular, to attain quick adaptation, some systems try to select between more than one modelling and customisation methods with different degrees of complexity. The choice depends on the amount and quality of data available.

1.4.3 Application domains and use in knowledge management

The application domains of user modelling include:
- educational and tutorial systems: knowledge and skill diagnosis, student modelling, tailoring to learner proficiency, feedback and support for collaboration,
- adaptive information filtering, retrieval and browsing: information retrieval (hypermedia navigation, intermediaries and information filtering), information presentation (decision support, dialog management, natural language interpretation, processing and generation),
- multi-modal user interactions and use of the model to integrate interactions and to interpret the user,
- supporting inter-user collaboration,
- e-commerce applications (acquiring user preferences, services and product customisation, profile management, automatic suggestion, shopping and comparison systems, etc.),
- consumer guides,
- mobile systems,
- interface adaptation (tailoring to abilities, disabilities, and preferences; provision of help; high-level programming and control),
- etc.

The beginning of a commercial boom of this domain in the late 1990s, was essentially due to the value of web customisation [Kobsa, 2001].

From the knowledge management perspective, adaptive information filtering and interfaces, collaborative software and tutorial tools applications are extremely close. Knowledge engineering is looking for tools capable of synthesising representations that best communicate a concept to a targeted user. Users are not created equal and there are as many possible customisations and adaptations as users. [Fischer, 2001] explains that the universal access issues that underlines a lot of the above points, is to write software for millions of users (at design time), while making it work as if it had been designed for each individual user (at use time), to provide universal access for people

with different (dis)abilities and to adapt to different profiles (e.g. easy to understand and use without prior experience for novices *vs.* complex systems, but useful for professionals).

According to [Brusilovsky, 2001] adaptive hypermedia have two main parent areas (hypertext and user-modelling) and at least six kinds of application systems: educational hypermedia, on-line information systems, on-line help systems, information retrieval hypermedia, institutional hypermedia, and systems for managing personalised views in information spaces. The following extract illustrates very well my conviction of the strong link between knowledge management and user-modelling:

Saying the right thing, at the right time, in the right way:

"The challenge in an information-rich world (in which human attention is the most valuable and scarcest commodity) is not only to make information available to people at any time, at any place and in any form, but to reduce information overload by making information relevant to the task-at-hand and to the assumed background knowledge of the users. Techniques to say the right thing include: (1) support for differential descriptions that relate new information to information and concepts assumed to be known by a specific user; and (2) embedded critiquing systems that make information more relevant to the task-at-hand by generating interactions in real time, on demand, and suited to individual users as needed. They are able to do so by exploiting a richer context provided by the domain orientation of the environment, by the analysis of partially constructed artefacts and partially completed specifications. To say things at the right time requires to balance the costs of intrusive interruptions against the loss of context-sensitivity of deferred alerts. To say things in the right way (for example by using multimedia channel to exploit different sensory channels) is especially critical for users who may suffer from some disability." [Fischer, 2001]

As I said about organisational modelling. The model I envisaged aimed at supporting corporate memory activities involved in the application scenario; I shall not discuss improvements in user modelling. My interest in user model techniques here is their direct use in order to enable the system to get insight in its environment (user profiles, communities of users, existing interests,...); to give the system an awareness of the users it is interacting with.

1.5 Information retrieval systems

Knowledge is of two kinds. We know a subject ourselves, or we know where we can find information upon it.
— Samuel Johnson

Since information is tightly linked to knowledge, information systems are tightly linked to knowledge management. Accessing the right information at the right moment may enable people to get the knowledge to take the good decision at the right time. Retrieval is the main issue of information retrieval systems, and I shall briefly survey different techniques developed in that field using the states of the art of [Greengrass, 2000] and [Korfhage, 1997]. I shall not enter into the details: there are whole books on just sub-fields of this subject (databases, information retrieval, data mining, information extraction, etc.), enough to fill many shelves in a library. Therefore, I shall only survey the existing lines of research to position my work and identify the different influences.

Ideally, an information system would somehow understand the content of documents and produce a single, concise, coherent and comprehensive answer relevant to the user's request. In actual fact, such a strategy is far beyond the state of the art [Greengrass, 2000].

Most of the research focused on textual documents since natural language remains the preferred form for knowledge communication. Thus information retrieval matches information to information need by matching documents to a query; information filtering proposes techniques for rapid selection of rich ore of document [Korfhage, 1997]; data mining and information extraction try to extract conceptual structures respectively from databases or documents.

Information retrieval often addresses the retrieval of documents from an organised and relatively static collection, also called archive, corpus or digital library. However it is not restricted to static collections and it can be applied to streams of information resources, (e.g. e-mail messages, news letters). Following [Korfhage, 1997], I shall distinguish between:
- *Retrospective search system*, designed to search the entire corpus (set of textual documents) in response to an ad hoc query. Such a system has a relatively large and stable corpus and a small, rapidly changing set of submitted queries, provided anew each time by the user.
- *Monitoring search system*, designed to keep its users informed concerning the state of the art in their areas of interest as specified in their profiles. These interest descriptions are run as routing queries against new documents, and the results are disseminated to users for whom there is a match. Such a system has a flow of documents, a relatively unstable corpus, and a large static set of queries derived from the user profile.

This distinction between routing and information retrieval is an idealisation. In practice, the distinction is not necessarily so clear-cut. Routing and information retrieval may be viewed as opposite ends of a spectrum with many actual applications in between. [Greengrass, 2000]

The storage problem of information systems will not be addressed here since it is not directly relevant to the my work; it is however a non negligible problem in a complete solution.

Thus a typical information retrieval system seeks to classify the documents of a given collection to find the ones that concern a given topic identified by a query submitted by the user. Documents that satisfy the query with regard to the judgement of the user are said to be 'relevant'.

Information retrieval generally focuses on textual resources. Other kinds of information resources such as images, sound, video, and multimedia documents combining several of these could be relevant for a query. However textual documents are the most current support of information and moreover, mining unstructured multimedia documents is extremely complex, even more than mining textual resources that is already a challenge as we shall see it through the following sections.

1.5.1 Resources and query: format and surrogates

1.5.1.1 Initial structure

An information resource such as a document, is structured if it consists of named components, organised according to some well-defined syntax and with a fixed meaning for every component of a given record type (e.g. database schema). This does not mean that the associated semantics is unambiguously explicit to anyone looking at the structure. Information resources may be structured, unstructured, semi-structured, or a mixture of these types.

In a solution of fully structured information such as in databases or knowledge-based systems, there exist fixed schemata, conceptual structures with known fields and logical rules. As long as the specifications of the data structure correspond to the needs, exact or range matching operators can be designed for each data type (e.g. for integers: =, >, ≤, etc.) and used to propose powerful retrieval algorithms. A search engine applies these operators to find a given component in a given structure and retrieve its contents. Similarly, given a component and a value, the search engine can find records such that the given component contains the given value. These approaches will encounter problems with imprecise or highly heterogeneous data and require to formalise all the information corpus in a logical format which can be impractical.

In a collection of unstructured resources such as texts in natural language, there is no unique and fixed syntactic position: in a random collection of documents, there is no guarantee that they are about the same topic, there is no guarantee on the information they specify or where it is specified. Unstructured means that there is no (externally) well-defined syntax shared by the documents. Even if a syntax is known to exist, the semantics of the syntactic components is unknown [Greengrass, 2000].

As usual there exist hybrids between these extremes, I shall distinguish two of them:

- *Informally structured documents (usually called semi-structured)*: the structure does not rely on a formal language. A collection of textual documents may share a common structure not completely marked out or explicit, but in these semi-structured documents, at least some information is located at a fairly standard position in the text with clues such as domain dependent keywords (e.g. "age:", "sex:", "population:" etc.) or document nature-dependent structural information (e.g. strategic position and formatting for the title, keywords such as "abstract:", "index", etc.) This makes it possible to write algorithms or at least heuristics for extracting the data back into a structured format. A classical example of resources that can be mined this way are the content of the pages of the CIA Fact Book. Ironically, semi-structured documents are usually the results of the front-end of a database or another archiving system; XML could be an asset in removing this unseemliness.
- *Hybrid structure, mixed structured or partly structured resources*: they include structured parts and unstructured ones sometimes intertwined e.g. Web pages have a structured header and an unstructured body. Structured part typically contains metadata (*i.e.*, data about the resource) rather than the information content of the resource itself, e.g. the author, title, recipient, abstract, creation date, keywords, ISBN and other id numbers, etc. With a language like XML language where text and tags can be mixed at will, a document can be anywhere between the two extremes.

The initial structure or the information resources varies on a continuum between unstructured resources, informally structured or semi-structured resources, hybrid partly-structured resources and strongly structured resources.

In any case, information retrieval focused a lot on searching the remaining unstructured part with the strong constraint of producing automatic and scalable methods and tools.

1.5.1.2 Surrogate generation

1.5.1.2.1 Surrogate for indexing and querying

One could envisage to scan the full collection of documents each time a query is issued. For instance a finite automata could be applied to search all the documents for a given string. In that case, no additional space than the original corpus is required and new document insertion is straightforward. This may be necessary if no restriction is placed on the form of queries that can be submitted, however it is a very slow approach because the whole corpus has to be scanned every time.

The key problems are how to state an information need and how to describe an information resource so as to be able to match the need to the resource. To tackle these problems, there are two major categories of approaches: semantic and statistical. Semantic approaches attempt to implement syntactic and semantic analysis to capture in semantic structures some aspects of the understanding of the resource that a human would have. In statistical approaches, the resources selected for retrieval or clustering are those that match the query most closely in terms of some statistical measure. However it is clear that each approach tends to introduce techniques from the other; statistical measures need to be attached a minimum of meaning in order to be usable and semantic features are reintroduced in the statistical spaces while semantic approaches use statistical tools to filter the corpora before processing thus improving their scalability.

There are two spaces, often with the same features, in nearly every approach: the document space and the query space. The *document space* is where documents are organised in order to prepare their use. Without this pre-organisation of documents, the query processing rapidly becomes prohibitively expensive; on the other hand, the pre-organisation must be designed so as to be adequate to the envisaged processing and the richer this pre-organisation is, the more likely it will require heavy pre-processing that will raise problem of scalability and costs of modification such as adding or removing documents or features; iterative construction methods are a must. The task of organising documents is called *indexing* and the resulting structure is the *index*. The first and simplest index type is the signature file: it consists of a sequence of document references and document signatures *i.e.,* bit strings derived from the document using hashing on its words. Indexes are used to create *inverted indexes i.e.,* indexes where the entrance points are the features used for indexing and the output is a list of candidate documents. For instance for the signature file, the inverse index would give for each word the documents that use it.

Indexing consists of scanning the corpus to build representative surrogates for each one of the information resources. This is the first facet of the use of surrogates in information system (the second one will be presented in the section about user interaction): the use of this surrogate (e.g. a vector) is to represent the resource in the processing carried out by the system (e.g. matching a query); it influences the whole system in its process (since the structure that feed the algorithms and the algorithms that exploit the structures are tightly linked), in its use (since if a surrogate eliminates an information -e.g. the order of terms- it will not be possible to use them for processing) and in its performances (since complex structure may provide finer results, but need much more computational power). Surrogates for indexing may be as simple as collecting some words of the document or as complex as a natural language analysis of its content.

The choice of the surrogate depends on the information coding and structure. In fact, I compare the surrogate coding and structure to a sort of highly degrading compression preserving only features relevant to information processing. Surrogates do not limit to the representation of the content, but can also include metadata (editor, author, ISBN, etc.) The whole set of these surrogates and its structure form the index of the corpus.

The first use of information resource surrogates is to provide a highly synthetic and representative structure that reduces the content of the resource to those features relevant for the intended processing.

Document granularity (a whole encyclopaedia, a volume, a chapter, a section, a page, a paragraph) and documents within documents raise the problem of choosing cut-point *i.e.,* on which partition will the surrogate calculation be based. This choice influences the relevance evaluation process: the parts of a document that has been split may not be individually relevant to a query, but collectively they may be; conversely, a whole encyclopaedia most probably contains a lot of answers, but for each one of them, only a short extract is most probably sufficient and really relevant. Document format is also a problem (text, image, sound, schema, table, application, data base file, etc.) and other characteristics (e.g. ephemeral documents *vs.* flow of documents *vs.* static collection, etc.).

On the other side of the coin is the *query space*. The forms of a query range from strongly constrained formal structures efficient for computational processing, but may be difficult to use for humans, to natural language queries that put a heavier computational load on the system, but are much more easily stated by the user. The methods for matching a document to a query are closely related to the query form used. The classic range of form is divided into the statistical approach and the semantic approach. The choice of the query space is not separable from the choice of the document space since elements from the query space and document space will have to be matched. A natural language query can be treated in a crude way, by removing stop-words, stemming other words, thus creating a list of terms. At the other extreme, the query undergoes a lexical, syntactic, semantic and pragmatic analysis. However, a query is usually much shorter than a text, thus it is quicker to process, but there are much less context and clues to interpret. As we shall see in the interface section, one can use a dialogue system to refine the query elicitation, a bit like it would happen if you were asking for information to a librarian.

An indexing and querying vocabulary has to be chosen (words, concepts, sentences, complex structures, etc.). A special constraint is the use of controlled vocabulary (*vs.* uncontrolled vocabulary) by surrogate authors and/or query authors. Controlled vocabulary obliges user to chose among a reduced, precise set of terms to express a document or a query; it is constraining, restricting, and sometimes frustrating, but it allows much more efficient processing. With an uncontrolled vocabulary the user is free to use any term, but it introduces risks of mismatch and losses because different indexing terms were used for describing documents and queries; it may also lead to huge indexing structures, however it allows for much more flexible solutions. As usual, the current trend is to mix both approaches, trying to allow a maximum flexibility at the user-end and implementing (semi)-automatic reduction algorithm to translate informal inputs into controlled (formal) and semantically richer structures. The translation usually rests on thesauri providing standard terms to index or query and cross-referencing relations such as "see also", "other", "broader term", "narrower term", "synonyms", "term co-occurring". Algorithms exist to automatically build a thesaurus by analysing co-occurrences of terms. We shall also see that one form that can be taken by a controlled vocabulary is called an *ontology*.

"Two characteristics of an indexing language are its exhaustivity and specificity. *Exhaustivity* refers to the breadth of coverage of the index terms - the extent to which all topics and concepts met in a document set are covered. *Specificity* refers to the depth of coverage - the extent to which specific topic ideas are indexed in details." [Korfhage, 1997]

The indexing may be manual or automatic:

- *manual indexing* relies on human interpretation and thus benefits from a better assessment of the content and topics of a document, but it is less systematic and therefore, bias and error prone. However controlled vocabulary assisted by semi-automatic annotation assistance can reduce these problems.
- *automatic indexing* usually uses term frequency in a document as an indicator of importance in the document and term average frequency occurrence in the collection as an indicator of its power of differentiation; it also relies on stemming and stop-term removal as pre-processing.

1.5.1.2.2 Statistic approaches to surrogate generation

In automatic indexing, the statistical approaches break documents and queries into terms that provide a population statistically counted and studied. Terms can be simple words, complete phrases recognised by frequency of co-occurrences or using dictionaries or, in the purest statistic approaches, *n*-grams *i.e.,* strings of *n* consecutive characters obtained by moving a window of *n* characters in length through a document or query, one character at a time [Greengrass, 2000]. Using *n*-grams is language-independent, and can even be used to sort documents by language. Techniques based on *n*-grams appear to be relatively insensitive to degraded text such as spelling mistakes, typos, etc. However the original structure of the natural language is completely lost and syntactic or semantic level techniques can hardly use the obtained index; the complete processing is purely statistic. Keyword indexing removes the structure and ordering which may be useful to introduce basic constraints in querying. An extension of keyword indexing is called full-text indexing and enables the system to preserve the notions of ordering and neighbourhood. Each term is indexed with its positions into the documents where it appears and thus enabling more expressiveness in queries.

The first and simplest surrogate is the boolean vector. A document is represented by a vector comprising 0 or 1 values respectively indicating for each term its absence or its presence. A document is represented by a set of its terms, phrases, or n-grams which sacrifices all the syntactic information about the order and structure in which the terms occur in the document. The query is formulated as a boolean combination of terms using classical operators: AND, OR, and NOT. Absence or presence of a term is a poor way of representing the relevance of a document and the first step to mitigate the qualification is the extended boolean form using weights or fuzzy logic approaches. Fuzzy set and fuzzy queries especially try to address the problem of non binary attributes.

The most common weighted form is the vector space where the document surrogate is the term vector in the document space. The union of all the sets of terms obtained by scanning the corpus defines the document space as a space in which each distinct term represents one dimension. In this space, a term vector represents a document by assigning a numeric weight to each term from the document, trying to estimate the usefulness of the given term as a descriptor of the given document. [Greengrass, 2000]. The weights assigned to the terms of a document are interpreted as the coordinates of the document in the document space (point or a vector from the origin). Conversely, the inverse of the document space is the *term space*, where each document is a dimension. Each point (or vector) is a term in the given collection the coordinates of which are the weights assigned to the given term in each document in which it occurs [Greengrass, 2000]. In a document weights measure how effective the given term is likely to be for distinguishing this document from another; in a query, they measure how much importance the term is to be assigned in computation of the matching of documents to the given query.

Weights are assigned to terms following statistical methods. The most famous one being "*term frequency in a document * inverse term frequency in the collection of documents*" commonly abbreviated "tf*idf". High-idf terms tend to be better discriminators of relevance than low-idf terms. However a lot of alternative weighting schemes have been proposed to offset some drawbacks, but there is no use to detail them here.

Weights undergo normalisation, the most common ones being the frequency and Euclidean normalisation. Normalisation of the term frequency, makes it depend on the frequency of occurrence relative to other terms in the same document, not its absolute frequency of occurrence; weighting a term by absolute frequency would favour longer documents over shorter documents. Euclidean normalisation divides each component by the Euclidean length of the vector. Here again alternative normalising techniques were proposed, aimed at factoring out the effects of document length.

The terms often undergo pre-processing. The most common pre-processes are the *stemming* and *stop-words removal*. A word is stemmed when the system extracts and only keeps its root. The goal is to eliminate the variation of different grammatical forms of the same word, e.g. "retrieve", "retrieved", "retrieves" and "retrieval". Stop-words are common words that have little power to discriminate relevant from non-relevant documents, e.g. "the", "a", "it", etc. To be able to operate the removal, surrogate generators are usually provided with a "stop list" *i.e.,* a list of such stop-words. Note that both stemming and stop lists are language-dependent [Greengrass, 2000]. Stemming and stop-words removal are the most common types of normalisation in traditional information retrieval system. They are also examples of introduction of some natural language processing in statistics techniques, even if they are quite low level techniques.

Concerning the pre-processing, [Greengrass, 2000] gives examples of higher level problems due to such low-level treatment: "assassination" (singular) in a document proved a much more reliable indicator that the document describes an assassination event than the presence of the term “assassinations” (plural) that often refers to assassinations in general. Likewise "venture" by itself is not a good indicator that a document describes a joint venture, but the phrases "venture with" and "venture by" (with the stop words) proved to be very good descriptors.

Vector spaces cannot address the problem of expressing logical connectivity as in a boolean query; to overcome this problem, extensions based on weight manipulation have been proposed.

Another major limitation of the vector space approach is that it assumes that terms are independent enough to provide orthogonal dimensions for the document space *i.e.,* relationships among the terms are ignored (co-occurrences, synonyms, etc.).

An alternative vector space approach, called *Latent Semantic Indexing* (LSI), attempts to capture these term-term relationships using statistics of co-occurrences. The LSI document space is a much lower dimensional space in which dimensions are artificial concepts statistically independent; these artificial concepts are sets of terms that are related (co-occurrence, synonyms, etc.). Thus documents and queries dealing with the same topic that could be far apart in traditional term-based document space because they use synonyms terms, may be close together in this space. This representation requires storage of a substantially larger set of values. Moreover, the technique is not a semantic method as pretended in its name, but rather a statistical method for capturing term dependencies that it is hoped have semantic significance; co-occurrences techniques tend to build lexical fields rather than synonym sets and the choice of the size of this set influences the results and precision of the retrieval techniques.

1.5.1.2.3 Semantic approaches to surrogate generation

Statistical methods are highly scalable and highly autonomous, but encounter problems inherent to the lexical level such as homographs (the converse problem of synonyms) that require syntactic, semantic and pragmatic analysis. They rely on the presence of a term in a document as if it meant the document is relevant to that term while it may have been have been merely mentioned in passing. As an example inspired by [Korfhage, 1997], if this PhD thesis includes the sentence. *I shall not talk about planning algorithm here because they are not directly relevant to this work*. It is clear that a term driven retrieval system may select it while it is literally not relevant...

Unlike statistical approaches, non statistical approaches are usually extremely interested in the structure; they are usually grouped under the appellation *natural language processing* (NLP). They attempt to address the structure and meaning of textual documents directly, instead of merely using

statistical measures as surrogates. Natural language processing may include: lexical analysis, syntactic analysis, semantic and pragmatic analysis. The output is usually a rich complex formal structures to be exploited in the retrieval process such as the conceptual graphs and the associated operators [Sowa, 1984].

[Greengrass, 2000] picked out several levels in natural language processing:
- *phonological level*: for speech recognition. It implies analysis of sounds the details of which are out of the scope of this work.
- *morphological level*: recognise the variant forms of a given word in terms. It is used to generate stemming systems; it is the first process of natural language processing tools before tagging words.
- *lexical level*: recognise the structure and meaning at the word level. It is used to build stop-words lists or thesauri, but also to detect and tag the words with their type such as proper noun, verb, etc. Proper nouns are excellent indicators of the relevance of a document. Their use may require common knowledge for instance if a document mentions Paris, then it matches queries about French towns. Proper nouns provide fixed point to resolve co-references if varied forms can be mapped into a standard form. (e.g. "President of United States in 1997", "Mister Clinton", "Bill Clinton", "Pres. Clinton", etc.)
- *syntactic level*: analysis of the structure of sentences. It can be used, for instance, to map passive forms to active forms, participating in the normalisation of the form of the sentence before semantic interpretation.
- *semantic level*: interpret meaning of the lower level results. It is used for disambiguation and construction of conceptual structures. Disambiguation can rely for instance on analysis of the local context of the term occurrences matching it to a thesaurus including the different meanings and their usual context.
- *discourse level*: interpret the document structure to determine the structure of clauses, sentences, and paragraphs that determines the rhetorical flow of a document and its meaning. It can be used for document summarisation or to type the knowledge extracted from a part.
- *pragmatic level*: introduce external knowledge such as the user profile, the context, common knowledge, domain knowledge to improve interpretation.

Besides the text analysis, surrogates can exploit other structural clues in the document that pertains to pragmatic knowledge of document structuring and domain knowledge. Examples of key indicators are:
- *bibliographic citations*: co-citation *i.e.,* co-occurrence of documents in citations of a group of documents on a subject, bibliographic coupling *i.e.,* two documents citing the same third document dealing with a given topic.
- *hyperlink*: the anchors of the link enable us to locate a set of related documents, and the labels of the links enables us to improve the description of the topics of these documents.
- *structural clues*: trigger terms such as 'legend', 'conclusion', 'example', etc. to locate important information in the text, source of document (type of journal, recognised author, etc.), etc.

1.5.2 Querying a collection

The problem is the evaluation of the topicality of the document [Korfhage, 1997] *i.e.,* how well the topic of the document matches the topic of the query.

Whatever model is used, ultimately the system calculates a function or a measure to evaluate the relevance of a document to a query. If a query is considered as a document then a similarity approach will most probably be applied; otherwise, the mapping of a document to the query or the projection of the query on the documents will use a relevance measure that can give a binary result (relevant / not relevant) or a degree of relevance (a percentage).

If the document space and the query space are based on terms, then the matching algorithm as to be designed to retrieve document that include the terms (some or all of them) of the query.

In a boolean space, the system has prepared a signature file for each document and evaluates the truth value of the boolean query represented by an expression composed of terms as boolean variables and boolean operators AND/OR/NOT, against each signature of the documents that fixes the value of each variable representing a term. For efficiency the query may be recast in more convenient forms for processing (disjunctive normal form, conjunctive normal form); they have an equivalent truth table while enabling optimised and simplified processing algorithms. Possible refinement for boolean query solving include: focusing on specified syntactic component of each document (the title, the abstract, etc.) or regions (e.g. beginning of the document); addition of operators, the most common one being the proximity operator forcing two keywords to be within a close distance in the text.

In the extension of boolean space by fuzzy logic, extended boolean operators make use of the weights assigned to the terms to evaluate their arguments. The result is no longer boolean, but a value between 0 and 1 corresponding to the estimated degree to which the given logical expression matches the given document.

We have seen that statistic methods are used to provide the weights of probabilistic queries; the membership is considered to be a probability thus between 0 and 1. The term probabilities are combined to calculate probabilities of a document to match a query *i.e.,* probabilistic approaches are trying to calculate the conditional probability $P(D|A, B, C, ...)$ that the given document D is relevant, given the clues A, B, C, etc. Making some statistical simplifying assumption, the conditional joint probability is replaced by a separate probability for each event. The usual assumption is that these properties are not independent, but that the same degree of dependence holds for both the relevant document set and the non-relevant document set, yet we have seen it was not exactly the case.

The vectorial representation leads to two techniques:

- *metrics or dissimilarity measures*: calculate a distance between the query point and the document point in the vector space or between two document points for clustering. Multiple reference points can also be used by the system: query, profile, other documents, known authors, known journal etc. They can be uses to specify the interest area and ask for document similar to these points. This is close to case-based reasoning and a lot of metrics have been proposed here too.
- *angular measures*: calculate the angle between the vectors representing the query and the document, or two documents for clustering for instance. The most famous one is the cosine measure. One problem with cosine similarity, is that it tends to produce relatively low similarity values for long documents, especially when the document is long because it deals with multiple topics. Thus here again alternative approaches and measures have been proposed.

Other probabilistic methods used are Bayesian probability and Bayesian networks, inference models, etc. they will not be developed here.

Once the optimised query is executed to the user's satisfaction, it is typically thrown away. An idea is to reuse queries and their results if there is a reasonable assumption that user information needs will recur; such queries are called *persistent queries*. If a new query is equivalent to a persistent query, the result is reused, otherwise the closest query is used a the start point of the search process [Greengrass, 2000]. I believe the use of persistent queries as representing persistent needs of communities of interest can be exploited even further in managing knowledge needs and fostering exchanges for monitoring a state of the art.

Another approach is the *clustering of documents* mainly based on statistical methods or graph theory. Clustering is the grouping of documents into distinct classes according to the properties captured in their surrogates. Clustering algorithms seek features that will separate the documents into groups ideally completely separate and as far apart as possible in feature space. It uses a document-document similarity and usually a threshold. Most of the algorithms iteratively build a hierarchy of clusters. Of course, and once again, it poses the problem of finding a reliable similarity and of choosing an adequate threshold. To group similar documents in clusters has a first interest linked to the previous search method: the acceleration of the searching of relevant documents; for instance, the algorithm compares its query to the centroids (virtual or real document surrogate representing the average profile of a cluster) of the clusters to determine in which cluster it is likely to find the best answers. Clustering documents within a collection is a form of unsupervised classification. Just as it can help searching algorithms to focus on the right cluster, it can also be used to provide browsing approaches. In *browsing*, the user starts searching the data without a clear-cut end goal in mind, without clear-cut knowledge of what data is available, and very likely, without clear-cut knowledge of how the data is organised. Clustering can reveal the intrinsic structure of a collection and when combined with user-friendly display, can be an effective tool for browsing a large collection and "zeroing in" on documents relevant to some given topic or other criterion. [Greengrass, 2000]

Going further than information retrieval that retrieves documents relevant to a given query or topic, information extraction tries to directly return the information needed by the user in a response that may have been generated from multiple resources. This area heavily relies on natural language analysis to understand the query, understand the documents and build an answer; many problems remain to be solved in this area.

Most of logic-based information retrieval systems are turning into web-based systems as part of the general trend to try to build a knowledge-based Web, or Semantic Web. In chapter 3, we shall see some examples of reasoning at the knowledge level to improve information retrieval. Indeed, if the surrogates are highly structured and exploit knowledge modelling languages, additional inference capabilities can be added to the improve retrieval, reasoning on models of the domain, deducing implicit knowledge, etc.

1.5.3 Query and results: the view from the user side

"A major factor in user acceptance of any retrieval system is the interface through which the user interacts with the system" [Korfhage, 1997]. Clearly there is a need for the development of visual information retrieval interfaces. Search results usually take the form of a list of documents ranked according to their estimated relevance to the query or topic for which they were retrieved. To represent this list, the system uses a second facet of document surrogates *i.e.,* the surrogate used to present and identify the document in front of the user. An identifier is always present in both types of surrogates, but it is not enough for user since it is usually a system identifier such as a URI http://www.mycorp.com/reportV278.htm#C12 or a key in databases. This second facet requires information such as: title, summary (that may require automatic generation methods), authors' abstract, focused extract, preview, snapshot, keywords, review, age, ISBN, position in a classification/presentation structure (e.g. in pop music, new releases, on sale, French speaking, etc.) etc. On the choice of the surrogate depends the ability of the system to propose views on the results that organise the selected document set so that the user gets the "big picture" quickly, zero in rapidly

on the desired documents, and see which documents are closely related. [Greengrass, 2000]. The bad news is that usually the relevance and choice of the content of a surrogate is domain-dependent.

The second use of information resource surrogates is to provide a highly synthetic and representative structure that reduces the content of the resource to those features relevant for the user to identify the resource and its content.

User interaction must not be reduced to the submission of a query; users interact with the system in many ways. They formulate queries, review the results, provide feedback, refine their original requests, describe their profile, build training sets, set algorithm parameters, etc.

When evaluating results, two positions can be adopted:
- *Quantitative assessment po*sition: the two main measures of information retrieval are recall and precision. Precision is the ratio of relevant items retrieved to all items retrieved. Recall is the ratio of relevant items retrieved to all relevant items available in the collection. Measuring precision is (relatively) easy if a set of competent users agree on the relevance or non-relevance of each of the retrieved resources. On the contrary, measuring recall is much more difficult because it requires knowing the number of relevant documents in the entire collection, which means that all the documents in the entire collection must be assessed. If the collection is large, this is quite simply not feasible [Greengrass, 2000].
- *Qualitative assessment position*: it tries to take into account the user's preferences and feedback such as the acceptance or rejection, the order of consultation, the novelty of documents, the number of documents examined before the relevant ones were found, the known reference documents found, the number of documents needed to get the answers *vs.* the number of document retrieved, etc. [Korfhage, 1997] also rightly distinguishes between relevance to the user's query, and pertinence to the user's needs.

A simple feature such as the size and grouping of the results can raise non trivial questions. For instance, on one hand, there may be multiple subsets of relevant resources, any of which will satisfy the user's requirement and on the other hand, two relevant resources may present contradictory views hence users would be seriously misled if they only see one of the resources.

One approach for dealing with the subjectivity of evaluation is to provide or generate "user profiles," *i.e.*, knowledge about the user's needs, preferences, etc. The objective is to give the user not just what he asked for, but what he "meant" by what he asked for. Or the profile may be generated automatically, based on statistics derived from documents the user has designated as relevant to his needs [Greengrass, 2000].

The user model is increasingly forming a significant component of complete solutions, introducing information about a user's preferences, background, educational level, familiarity with the area of inquiry, language capabilities, journal subscriptions, reading habits, etc. into the retrieval process [Korfhage, 1997]. There are several ways to enrich queries with profile information:
- *profile used as post-filter*: used after the query is processed to sort the results.
- *profile used as pre-filter*: modify the query before processing to focus the search.
- *profile used as co-filter*: documents are compared to both the profile and the query in a combining evaluation function.

Solutions including user profiles may use the multiple reference points mentioned before. Additional factors can also introduce additional retrieval tests such as "is the document too old?", "has it already been consulted by the user", "is it too detailed/shallow?", "is it written for/by an expert/novice?".

Another approach exploiting user profiles is *collaborative filtering*. It compares user profiles to build recommendations: if two user profiles are close, then it is likely that the consultation of one user can be used to make suggestions to the other, and vice versa. This fosters the emergence of communities of usage and communities of interest.

[Greengrass, 2000] calls *interactive directed searching* means, the systems in which the user engages in an interactive process, either to formulate the original query, or to refine the query on the basis of the initial results returned. A significant improvement in the performance of any solution is gained by involving the user in the query processing of refinement [Korfhage, 1997].

The process of query expansion and re-weighting may be wholly automatic or may involve a combination of automatic processes and user interaction. Relevance feedback in query expansion and refinement, is the classic method of improving a query interactively: based on the feedback, the system automatically resubmits or proposes expansions and refinements of user-generated queries.

In vector spaces, a classic refinement method is term re-weighting. Given a relevance feedback on a result, the system increases the weights of terms that occur in relevant documents and reduces the weights of terms that occur in non-relevant documents. It can also modify the document vectors by adding terms drawn from the user's query or application domain to the indexes of documents judged relevant, thus keeping a memory of terms judged relevant for the construction of a surrogate. Query expansion usually means addition of relevant terms for instance drawn from the most relevant documents or from thesauri giving synonymous terms.

1.5.4 The future of information retrieval systems

The future lies in hybrid solutions.

Hybrid documents: Most of the research focused on textual documents, but as multimedia documents are integrated in information systems, the ability to process and retrieve images and sounds is becoming important. Multimedia querying is in its infancy and most existing systems rely on descriptions of the multimedia resources, extracted information from the documentary context e.g. legend, surrounding text, etc. There exist experimental image retrieval systems using image recognition, classifier, etc. exploiting image features (colors, shades, outlines, etc.). Even less developed, sound retrieval systems also extract features (rhythm, musical patterns, etc.) to develop similarity measures and matching systems, classifier, etc.

Moreover, "many documents are in an intermediate or mixed form, largely unformatted, but including some formatted portions. This suggests that a two-stage retrieval process might be efficient - doing a rough retrieval based on the formatted portion of the data, then refining the ore generated by this process to locate the desired items" [Korfhage, 1997]. In hybrid systems where document and structured information are used, the retrieval can concern, both the knowledge contained in the document and the indexing structure (e.g. "Find documents that contain the words w_1,..., w_n written by M. XYZ with the name of the editor"). Depending on the users, their profiles and contexts, hybrid systems will handle differently mixed documents an mixed data (blob, large text fields, etc.). The available collections and their structure may themselves be semantically described to guide the choice of a searching algorithm for a given query and a given collection.

Hybrid systems: Information retrieval system solutions span many research fields, such as linguistics, databases, logics, etc. Each of them has its advantages and drawbacks, so the trend is to mix them in hybrid solutions to get the best of each. There are several ways of coupling different systems. The first manner is to cascade method complexity. For instance some system starts with coarse retrieval of candidate resources using statistical methods and shallow natural language extensions, then more sophisticated natural language processing tools are applied to the resulting list of resources retrieved by the first stage [Greengrass, 2000]. This technique allows quick filtering of information, refinement and focused passage retrieval, localising hot spots in a document where the interest is stronger. The other approach consist of merging results from parallel searches, such as it is done in meta-search engines. Fusion may be carried out between documents from different collections and / or using different search methods; in all cases different results must be compared to merge them into a single homogeneous result that can be presented to the user. Automatic adaptation of a method or a collaboration of methods may be done, using, for instance, genetic algorithms based on feedback. The introduction of 'recommender' systems based on collaborative filtering has the advantage of working for any type of resources and of introducing a collaborative improvement of the search mechanisms. The problem of multiple collections and systems meets the problematic of parallel and distributed systems, introducing new hard complexity concerns.

Finally, as information systems become accepted and used in our society, they raise new problems of ethical nature, legal nature (copyright, privacy), security nature, etc.

2 Ontology and knowledge modelling

"When I use a word," Humpty Dumpty said in rather a scornful tone,
"it means just what I choose it to mean - neither more nor less."
— Lewis Carroll

Knowledge engineering is a broad research where the overall issue is the acquisition and modelling of knowledge. Modelling knowledge, consists in representing it in order to store it, to communicate it or to externally manipulate it. Automating its external manipulation leads to the design of knowledge-based systems *i.e.,* systems which behaviour relies on the symbolic manipulation of formal models of knowledge pieces in order to perform meaningful operations that simulate intelligent capabilities.

The representation step raises the problem of the form *i.e.,* the choice of a representation formalism that allows to capture the semantics at play in the considered pieces of knowledge. One approach that emerged in the late 80s is based on the concept of ontologies. An ontology, as we shall see, is that part of the knowledge model that captures the semantics of primitives used to make formal assertions about the application domain of the knowledge-based solution.

Thus, in this chapter, I shall focus on the branch of knowledge modelling that applies logic and develops ontology to build computable models of some application domain. I shall divide my discussion in two large sections:
- the *ontology object*: focusing on the nature and the characteristics of the ontology object, its core notions and its lifecycle.
- the *ontology engineering*: focusing on the design rationale and the assisting tools.

2.1 Ontology: the object

"The word *ontology* comes from the Greek ontos for being and logos for word. It is a relatively new term in the long history of philosophy, introduced by the 19th century German philosophers to distinguish the study of being as such from the study of various kinds of beings in the natural sciences. The more traditional term is Aristotle's word category (kathgoria), which he used for classifying anything that can be said or predicated about anything." [Sowa, 2000b]

The word *ontology* can be used and has been used with very different meanings attached to it. Ironically, the ontology field suffered a lot from ambiguity. The Knowledge Engineering Community borrowed the term *Ontology* from the name of a branch of philosophy some 15 years ago and converted into an object: an *ontology*. In the mid-90s philosophers 'took it back' and began to clean the definitions that had been adopted. In this part, I summarise some of the survey I wrote in [Gandon, 2002a]. I shall focus on the definitional and design aspects that were of direct interest to my PhD.

2.1.1 Ontology: an object of artificial intelligence and conceptual tool of Knowledge Modelling

Ontology is a new object of Artificial Intelligence that recently came to maturity and a powerful conceptual tool of Knowledge Modelling. It provides a coherent base to build on, and a shared reference to align with, in the form of a consensual conceptual vocabulary on which one can build descriptions and communication acts.

"People, organisations and software systems must communicate between and among themselves. However, due to different needs and background contexts, there can be widely varying viewpoints and assumptions regarding what is essentially the same subject matter. Each uses different jargon; each may have differing, overlapping and/or mismatched concepts, structures and methods" [Uschold and Gruninger, 1996]. Some of the consequences of a lack of a shared understanding are: poor communication, difficulties in identifying requirements and therefore, in specifying a system, limited inter-operability, limited potential of reusability and sharing, therefore, wasted efforts in re-inventing the wheel. There is a need to "reduce or eliminate conceptual and terminological confusion and come to a shared understanding. (...) the development and implementation of an explicit account of a shared understanding (*i.e.,* an 'ontology') in a given subject area, can improve such communication, which in turn can give rise to greater reuse and sharing, inter-operability, and more reliable software" [Uschold and Gruninger, 1996]. An ontology is a unifying framework for different viewpoints and serves as the basis for enabling communication between people, between people and systems, between systems: this unifying conceptual framework is intended to function as a lingua-franca.

Ontologies are to semantics, what grounding is to electronic: a common base to build on, and a shared reference to align with. Ontologies are considered as a powerful tool to lift ambiguity: one of the main roles of ontologies is to disambiguate, providing a semantic ground, a consensual conceptual vocabulary, on which one can build descriptions and communication acts. [Bachimont, 2001] explains that on the one hand, ontologies provide notional resources to formulate and make explicit knowledge and on the other hand, they constitute a shared framework that different actors can mobilise. Ontology can represent the meaning of different contents exchanged in information systems.

The more we develop intelligent information systems, the more the general knowledge about things and their categories appears to play a pivotal role in inferences. Therefore, this knowledge needs to be given to the machines if we want them to behave intelligently and intelligibly. To illustrate that

point I shall take an example of problem where ontologies prove to be useful in the context of the work reported here: information retrieval. The general problem is to formulate a query over a mass of information and get an answer as precise and relevant as possible.

In her tutorial at ECAI 98, Assunción Gómez-Pérez asked the participants: "What is a pipe ?". Extending her example we can imagine four answers to this very same question and they are given in a dictionary definition:

> **PIPE** noun [C]. A short narrow tube with a small container at one end, used for smoking e.g. tobacco. || A long tube made of metal or plastic that is used to carry water or oil or gas. || A temporary section of computer memory that can link two different computer processes. || A simple musical instrument made of a short narrow tube which is played by blowing through it.

One term linked to four concepts is a case of ambiguity. The contrary is one concept denoted by several terms, and it is a case of synonyms e.g.

> **CALUMET, PEACE PIPE, PIPE OF PEACE** A highly decorated ceremonial pipe of Amerindians smoked on ceremonial occasions, especially as a token of peace.

These trivial cases pose a serious problem to computerised systems that are not able the see these differences and equivalences unless they have been made explicit to them. Indeed if we take the example of a classic user of the Altavista search engine looking for books by Ernest Hemingway, the commonly chosen keywords are "+book +hemingway". The search engine will encounter several types of problems:

- *Noise*: a problem of precision that will lead the search engine to collect a page with the sentence "The Old Book Pub, 3 Avenue Hemingway" while it is obvious to us that this is not relevant (unless one wants to drown one's sorrows of not having found the books).
- *Missed answer*: a problem of recall where the search engine misses a page containing a sentence such as "The novel 'The Old Man and The Sea' by Ernest Hemingway" because it does not know the basic categories of documents, and therefore, does not know that a novel is a book.

If we look at the way a human does answer a question, we may find interesting leads to solve the problem. Consider this little speech between two persons:

> "What is the last document you read ?" Rose asked.
>
> "The article Gruber wrote on ontology in 1993." Olivier answered.

The answer Olivier gave is based on an organisation of concepts used for at least two purposes:

- Identification: the ability to recognise an object, an action, etc. as belonging to a category e.g. the ability to recognise an object as being a book and an action as being reading.
- Specialisation and generalisation: the ability to memorise abstractions of categories differentiated in hierarchies of specialisation/generalisation e.g.: "articles, books, and newspapers are documents", "novels are books", etc. These hierarchies are the basis of inferences at the heart of information retrieval and exchange e.g.: the syllogism "a novel is a book" and "a book is a document" therefore, "a novel is a document" so if I am looking for a document, a novel is a valid answer.

This structure of categories is learnt through education and social cultural interactions. For instance imagine the following naive story:

> A family is on the road for holidays. The child sees a horse by the window, it is the first time he sees a horse.
>
> "Look mum... it is a big dog !" The child says.
>
> The mother looks and recognises a horse.
>
> "No Tom, it is a horse... see it's much bigger !" The mother corrects.
>
> The child adapts his categories and takes notes of the differences he perceives or he is told, to differentiate these new categories from others. A few kilometres later the child sees a donkey for the first time.
>
> "Look mum... another horse !" The child says.
>
> The mother looks and recognises the donkey.
>
> "No Tom, it is a donkey... see it's a little bit smaller, it is grey..." The mother patiently corrects.
>
> And so on.

In these interactions, categories are learnt, exchanged and aligned. This will enable understanding in the future when they will be used for communication.

Thus this structure in hierarchical categories captures a consensus and is socially and culturally dependent. If there is a mismatch or a lack, an interaction takes place to align the two opinions or fill the gap as in the example of the child. The consensus is implicit: in the case of the interactions about the document, both speakers implicitly consider that they have a shared and consensual conceptualisation of the reality of documents. By answering with *an article* the second speaker considers that the first speaker knows that *an article is a document*.

This background knowledge is lacking in information systems relying only on terms and plain-text search. A possible approach is thus to make this knowledge explicit and capture it in logical structures that can be exploited by automated systems. This is exactly the purpose of an ontology: to captures the semantics and relations of the notions we use, make them explicit and eventually code them in symbolic systems so that they can be manipulated and exchanged.

2.1.2 Definitions adopted here

I shall give here some of the latest definitions proposed in the knowledge engineering community and adopted in this work. I added personal definitions and dictionary definitions of notions commonly used in the field.

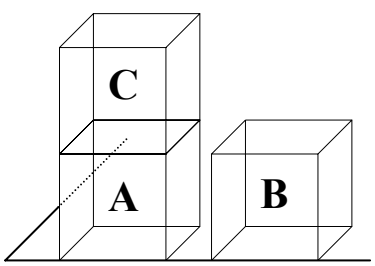
notion	something formed in the mind, a constituent of thought; it is used to structure knowledge and perceptions of the world. || an idea, a principle, which can be semantically valued and communicated.
concept	notion usually expressed by a term (or more generally by a sign) || a concept represents a group of objects or beings sharing characteristics that enable us to recognise them as forming and belonging to this group.
relation	notion of an association or a link between concepts usually expressed by a term or a graphical convention (or more generally by a sign)
extension / intension	distinction between ways in which a notion may be regarded: its extension is the collection of things to which the notion applies; its intension is the set of features those things are presumed to have in common. There exists a duality between intension and extension: to included intensions $I_1 \subset I_2$ correspond included extensions $E_1 \supset E_2$.
concept in intension / intension of a concept	set of attributes, characteristics or properties shared by the object or beings included in or to which the concept applies. e.g. for the concept of a *car* the intension includes the characteristics of *a road vehicle with an engine, usually four wheels and seating for between one and six people*.
concept in extension / extension of a concept	set of objects or beings included in or to which the concept applies. e.g. for the concept of a *car* the extension includes: *the Mazda MX5 with the registration 2561 SH 45*, *the green car parked at the corner of the road in front of my office*, etc.
relation in intension / intension of a relation	set of attributes, characteristics or properties that characterises every realisation of a relation. e.g. for the relation *parenthood* the intension includes the characteristics of *the raising of children and all the responsibilities and activities that are involved in it*.
signature of a relation	set of concepts that can be linked by a relation, this constraint is a characteristic of the relation that participate to the definition of its intension. e.g. for the relation *parenthood* the signature says it is a *relation between two members of the same species*.
relation in extension / extension of a relation	set of effective realisations of a relation between object or beings. e.g. for the relation *parenthood* the extension includes: *Jina and Toms are the Parents of Jim*, *Mr Michel Gandon is my Father*, etc.
Ontology	that branch of philosophy which deals with the nature and the organisation of reality [Guarino and Giaretta, 1995]. || a branch of

	metaphysics which investigates the nature and essential properties and relations of all beings as such.
Formal Ontology	the systematic, formal, axiomatic development of the logic of all forms and modes of being [Guarino and Giaretta, 1995].
conceptualisation	an intensional semantic structure which encodes the implicit rules constraining the structure of a piece of reality [Guarino and Giaretta, 1995] || the action of building such a structure.
ontology	a logical theory which gives an explicit, partial account of a conceptualisation [Guarino and Giaretta, 1995] (based on [Gruber, 1993]); the aim of ontologies is to define which primitives, provided with their associated semantics, are necessary for knowledge representation in a given context. [Bachimont, 2000]
ontological commitment	a partial semantic account of the intended conceptualisation of a logical theory [Guarino and Giaretta, 1995] || practically, an agreement to use a vocabulary (*i.e.,* ask queries and make assertions) in a way that is consistent with respect to the theory that specifies the ontology. Software pieces are built so that they commit to ontologies and ontologies are designed so that they enable us to share knowledge with and among these software pieces. [Uschold and Gruninger, 1996]
ontological theory	a set of formulas intended to be always true according to a certain conceptualisation [Guarino and Giaretta, 1995].
ontological engineering	the branch of knowledge engineering which exploits the principles of (formal) Ontology to build ontologies [Guarino and Giaretta, 1995]. || defining an ontology is a modelling task based on the linguistic expression of knowledge. [Bachimont, 2000]
ontologist	a person who builds ontologies or whose job is connected with ontologies' science or engineering.
state of affairs	the general state of things, the combination of circumstances at a given time. The ontology can provide the conceptual vocabulary to describe a state of affairs. Together this description and the state of affair form a model.
taxonomy	a classification based on similarities.
Mereology	The study of part-whole relationships.
partonomy	a classification based on part-of relation.

Table 2 Definitions used in ontology engineering

In order to express and communicate an intension, we choose a symbolic representation e.g. the different definitions associated to a term and given by a dictionary. Note that exemplification and illustration used in dictionaries show that it is sometimes necessary to clarify a definition in natural language, producing a representative sample of the extension (*i.e.,* examples) or using other means of representation (e.g. a picture).

To exemplify these definitions, we can reuse the well-known situation of cubes on a table. Figure 4 shows a schema depicting the real scene of three cubes being arranged on a table. A conceptual vocabulary (toy ontology) is proposed to talk about some aspects of this reality (some of them are ignored, for instance there is no vocabulary to express the dimensions of the cubes). Finally, the state of affairs of the scene observed is described using the primitives of the ontology.



Cube (X): The entity X is a right-angled parallelepiped with all its edges of equal length.
Table: A global object which is a furniture composed of an horizontal flat top put down on one or more legs.
On (Cube: X, Cube: Y / Table): a relation denoting that a cube X is on top of another Cube Y or on top of the Table

Cube (A)
Cube (B)
Cube (C)
On(A, Table)
On(C,A)
On(B, Table)

a - Reality **b** - Ontology **c** - Describe State of Affairs

Figure 4 The example of cubes

This example illustrates the definition of an ontology:

The ontology is...	**because...**
an explicit	the column b makes explicit the concepts used here.
partial account	some aspects were overlooked e.g.: the dimensions of the cubes, their material, etc.
of a conceptualisation	in this reality we recognise some entities (cube, table, etc.) and some rules (spatial relations, geometry, labelling of cubes etc.)

It is important to stress that nothing, in the original definition of an ontology used in knowledge engineering (*i.e.,* "an ontology is a specification of a conceptualisation" [Gruber, 1993]) nor in the definition we gave above, obliges the ontologist to use a formal language to make the ontology explicit. The representations of intensions can be organised, structured and constrained to express a logical theory accounting for relations existing between concepts; an ontology is an object capturing this theory. The final representation of the intensions and the ontological structure can make use of more or less formal languages, depending on the intended use of the ontology. An automated exploitation of an ontology by an artificial system will most probably imply some formalisation of some chosen aspects of the ontology to enable the formal manipulation of those aspects. The formal expression of an intension provides a precise and unambiguous representation of the meaning of a concept; it allows software to manipulate it and use it as well as the models the representation of which is based on the primitives provided by the ontology. [Sowa, 2000b] distinguishes between a terminological ontology and a formal ontology. They are the two extreme of a continuum: as more axioms are added to a terminological ontology, it may evolve into a formal or axiomatised ontology.

Symbols are at the base of our natural language and therefore, are the most commonly used means for communicating concept. They are also used to build artificial symbolic systems at the base of automation; the most advanced artificial symbolic systems are logical systems and other derived knowledge representation languages of all sorts. Thus ontologies are massive collections of Peirce's three kinds of signs: icons, which show the form of something; indices, which point to something; and symbols, which represent something according to some convention. [Sowa, 2000b]. For instance the concept of 'fire' can be represented by signs such as in Figure 5:

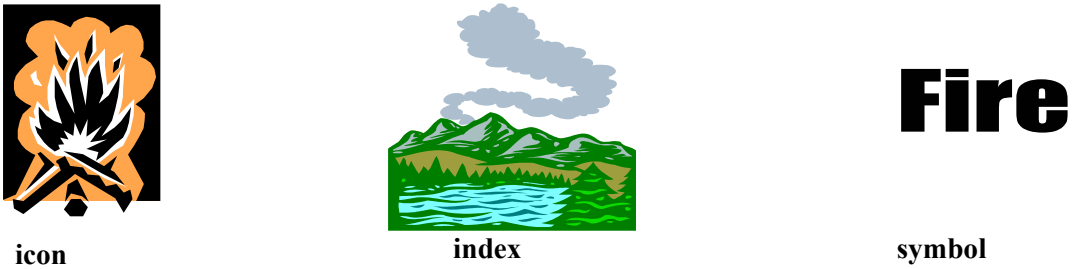
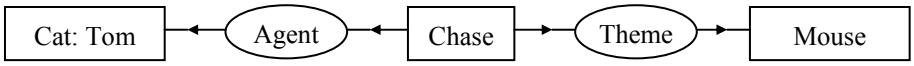


Figure 5 Icon, Index and Symbol

However, bare symbolic systems are ontologically neutral. [Sowa, 2000b] explained that an uninterpreted logic "imposes no constraints on the subject matter or the way the subject is characterised. By itself, logic says nothing about anything, but the combination of logic with an ontology provides a language that can express relationships about the entities in the domain of interest."

Sowa [2000] gives the following example represented here in logic and conceptual graphs [Sowa, 1984]. The fact represented here is "Tom the cat is chasing a mouse".

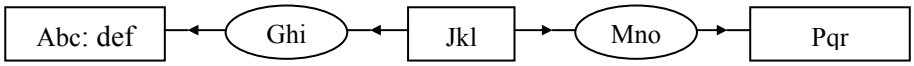
(∃ x: Cat) (∃ y: Chase) (∃ z: Mouse) (Identifier(x , "Tom") ∧ Agent(y , x) ∧ Theme(y , z))



This formula and the associated conceptual graph introduce several ontological assumptions because they suppose that there exist entities of types Cat, Chase, and Mouse; that some entities have character strings as names; and Chase can be linked to two concepts of other entities by relations of type Agent and Theme, etc. [Sowa, 2000]

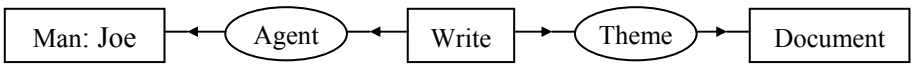
Now, to show that the logic does not capture the meaning of the primitives and that we interpret the meaning of the primitives because they are words of our natural language, let us consider the following formula and graph:

(∃ x: Abc) (∃ y: Jkl) (∃ z: Pqr) (Stu(x , "def") ∧ Ghi (y , x) ∧ Mno(y , z))



They are logically equivalent to the one of "Tom the cat is chasing a mouse", but without any interpretation possible because the primitives have lost the ontological meaning we were able to find using natural language. This very same formula could mean "The man Joe is writing a document":

(∃ x: Man) (∃ y: Write) (∃ z: Document) (Identifier(x , "Joe") ∧ Agent(y , x) ∧ Theme(y , z))



By maintaining human understandable representations of the notions, the ontology captures the mapping between the symbolic system used for artificial intelligence manipulation and the observations of the real world viewed from the perspective of an adopted conceptualisation. By capturing this isomorphism the ontologist intends to capture an advocated consensual interpretation for the users of the system.

Now what is the place / position of ontology in knowledge? If we consider the sentence "children are young humans", the context is not clear yet this is perfectly intelligible, perfectly usable, this is knowledge. In fact, the context here is general: it is knowledge about a category of things and not a about a specific occurrence of a concept. Thus this knowledge is universally verified in the human culture. This is typically an ontological piece of knowledge.

Taxonomy is one of the possible structuring for an ontology, it is a form of logical theory. Its importance comes from the fact that it supports elementary inferences constantly at play in information searching and communicating processes: identification and generalisation/specialisation. The previous example of ontological knowledge *children are young humans* could be positioned in a taxonomy as the one given in Figure 6.

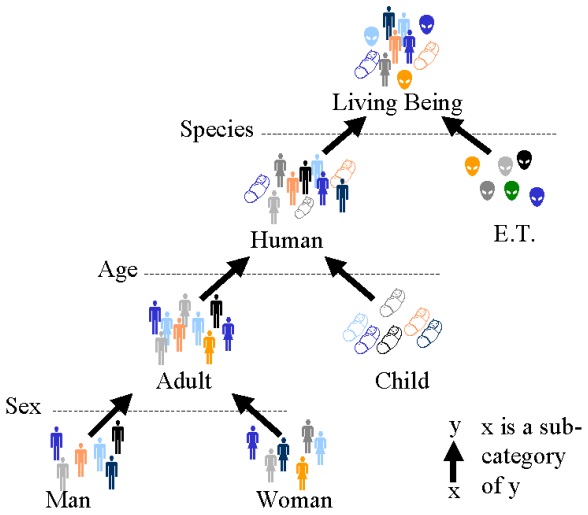


Figure 6 Simple taxonomy

But 'ontology' is not a synonym of 'taxonomy'. Other logical theories are useful to capture the definitional characteristics of the concepts we manipulate. For instance, the concepts of chemical elements make extensive use for their definition of partonomies: (water (H_2O) and phenol (-OH) contain hydrogen (H) and oxygen (O); alcohol (methanol CH_3-OH, Ethanol C_2H_6-OH, etc.) contain phenol, hydrogen and carbon etc.). An example of such structure is given in Figure 7.

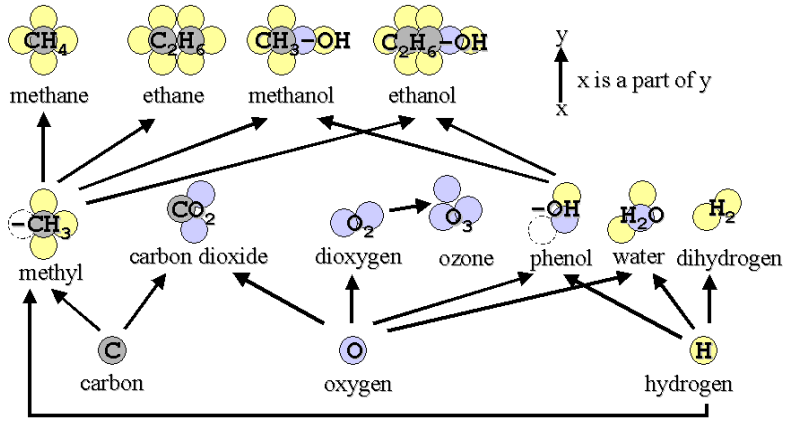


Figure 7 Simple partonomy

Partonomies are not the unique alternative to taxonomies. Further logical formalisations can be done especially to enable further inference capabilities such as automatic classification of new categories or identification of objects e.g.:

$$\text{director}(x) := \text{person}(x) \wedge (\exists y\ \text{organisation}(y) \wedge \text{manage}(x,y))$$

Or to capture causal models e.g.:

$$(\text{salty things} \Rightarrow \text{thirst}) \wedge (\text{thirst} \Rightarrow \text{to drink})\ \textbf{thus}\ (\text{salty things} \Rightarrow \text{to drink})$$

Sometimes instances are included in ontologies they could be called universal instances. For instance constants (e.g. *c* the speed of light, **g** the gravity constant,...) or global objects (the activity "the research") to enable a unique reference to the object. But this is not the real purpose of ontologies: ontologies concentrate on universal concepts and reify them if the need comes to use these concepts as objects of the discourse.

2.1.3 Overview of existing ontologies

The OntoWeb[1] European network has built a web application in order to allow the community to register their ontologies, methodologies, tools and languages for building ontologies, as well as applications in areas like: the semantic web, e-commerce, knowledge management, natural language processing, etc.

In order to witness the large number of application domains of ontologies, I shall give in the first following sub-section an overview of the variety of domains where ontologies were built; to do so I used the online report[2] mentioned above. In a second section I shall focus on the ontologies that are close to my subject, *i.e.,* organisations and knowledge management.

2.1.3.1 A variety of application domains

As we shall see, ontologies vary a lot in their form, exhaustivity, specificity and granularity. Thus some of the resources mentioned here are a thesaurus while other are formal logical theories.

- *AAT*: Art & Architecture Thesaurus to describe art, architecture, decorative arts, material culture, and archival materials.
- *Airport Codes*: simple ontology containing Airport codes around the world.
- *ASBRU*: provides an ontology for guideline-support tasks and the problem-solving methods in order to represent and to annotate clinical guidelines in standardised form.
- *Bibliographic Ontology*: a sample bibliographic ontology, with data types taken from ISO standard.
- *FIPA Agent Communication Language*: contains an ontology describing speech acts for artificial agents communication.
- *CHEMICALS*: Ontology containing knowledge within the domain of chemical elements and crystalline structures.
- *CoreLex*: an ontology for lexical semantic database and tagset for nouns, organised around systematic polysemy and underspecification.
- *EngMath*: mathematics engineering ontologies including ontologies for scalar quantities, vector quantities, and unary scalar functions.
- *Gene Ontology*: A dynamic controlled vocabulary that can be applied to all eukaryotes even as knowledge of gene and protein roles in cells is accumulating and changing. The three organising principles of this ontology are molecular function, biological process and cellular component.
- *Gentology*: genealogy ontology for data interchange between different applications.
- *Knowledge Representation Ontology*: from the book Knowledge Representation by John F. Sowa, this ontology proposes a top level for knowledge representation based on basic categories and distinctions that have been derived from a variety of sources in logic, linguistics, philosophy, and artificial intelligence.
- *Open Cyc*: an upper ontology for all of human consensus reality *i.e.,* 6000 concepts of common knowledge.
- *PLANET*: Planet is a reusable ontology for representing plans that is designed to accommodate a diverse range of real-world plans, both manually and automatically created.
- *ProPer*: ontology to manage skills and competencies of people.
- *SurveyOntology*: ontology used to describe large questionnaires; it was developed for clients at the census bureau and department of labour.
- *UMDL Ontology*: ontology for describing digital library content.
- *UMLS*: the Unified Medical Language System provides a biomedical vocabulary from disparate sources such as clinical terminologies, drug sources, vocabularies in different languages, and clinical terminologies.

[1] http://www.ontoweb.org
[2] http://babage.dia.fi.upm.es/ontoweb/wp1/OntoRoadMap/index.html

This shorten list of some existing ontologies allows me to clearly show that the application of ontologies covers a broad range of domains and that ontology engineering has already developed a set of varied ontologies that is increasing every days.

2.1.3.2 Some ontologies for organisational memories

Some ontologies have been studied in this work and were partially reused or influenced our choices; three of them are extremely known:

- *Enterprise Ontology*: The Enterprise Ontology is a collection of terms and definitions relevant to business enterprises. The ontology was developed in the Enterprise Project by the Artificial Intelligence Applications Institute at the University of Edinburgh with its partners: IBM, Lloyd's Register, Logica UK Limited, and Unilever. The project was support by the UK's Department of Trade and Industry under the Intelligent Systems Integration Programme (project IED4/1/8032). Conceptually, the Enterprise Ontology it is divided into a number of main sections: activities and processes, organisation, strategy and marketing.
- The *Dublin Core* Element Set Ontology: ontology interoperable metadata standards and developing specialised metadata vocabularies for describing resources that enable more intelligent information discovery systems.
- *TOVE*: The goal of the TOronto Virtual Enterprise project is to create a data model that provides a shared terminology for the enterprise, defines the meaning of each term in a precise and as unambiguous manner as possible, implements the semantics in a set of axioms, and defines a symbology for depicting a the concepts.

In addition, to the above ontologies, some existing ontology-based systems influenced my work; they will be presented in chapter 3.

2.1.4 Lifecycle of an ontology: a living object with a maintenance cycle

When an ontology participates to knowledge modelling as the one of a corporate memory, it becomes a full component of the model. It is also subject to its lifecycle since evolutions of the modelling needs may imply evolutions of the modelling primitives provided by the ontology. Therefore, the lifecycle of a corporate memory depicted in Figure 2 (page 33) also applies to an ontology that would be included in that memory.

The design of an ontology is an iterative maturation process. Through iterative design and refinement, we augment the ontology, developing the formal counter parts of semantic aspects relevant for the system in the application scenarios. This means the ontology will come to full development, becoming mature, by evolving through intermediate states to reach a desired state.

As soon as the ontology becomes large, the ontology engineering process has to be considered as a project, and therefore, project management methods must be applied. [Fernandez *et al.*, 1997] recognised that planning and specification are important activities. The authors give the activities to be done during the ontology development process (Figure 8): planning, specifying, acquiring knowledge, conceptualising, formalising, integrating, implementing, evaluating, documenting, and maintaining.

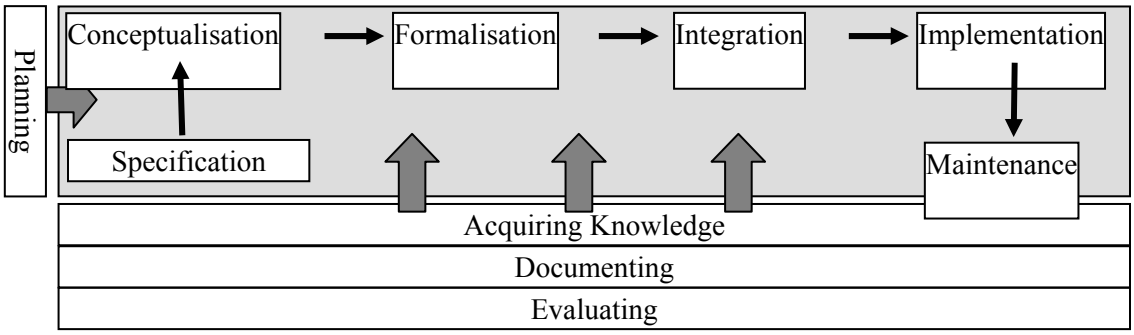


Figure 8 States and activities in the ontology lifecycle [Fernandez *et al.*, 1997]

[Fernandez *et al.*, 1997] criticise the waterfall and the incremental lifecycles and propose to use the evolving prototype lifecycle that lets the ontologist modify, add, and remove definitions in the ontology at any time if some definition is missed or wrong. Knowledge acquisition, documentation and evaluation are support activities that are carried out throughout these states. When applied to a changing domain, the ontology will have to evolve. At anytime someone can ask for including or modifying notions of the ontology. To maintain the ontology is an important activity to be done carefully.

Ontology design is a project, and should be treated as such, especially when it becomes large. Project Management and software engineering techniques and guidelines should be adapted and applied to ontology engineering. [Fernandez *et al.*, 1997] stress that before building your ontology, you should plan the main tasks to be done, how they will be arranged, how much time you need to perform them and with which resources (people, software and hardware).

Merging the two lifecycle diagrams of Figure 2 and Figure 8, I propose the cycle depicted in Figure 9. The Design & Build and Evolution activities are very close; they both include the activities of specification, conceptualisation, formalisation and implementation relying on data collected (knowledge acquisition or integration of whole or part of existing ontologies). Evaluation and Evolution respectively are the activities triggering and carrying out maintenance. Evaluation uses similar techniques to needs detection.

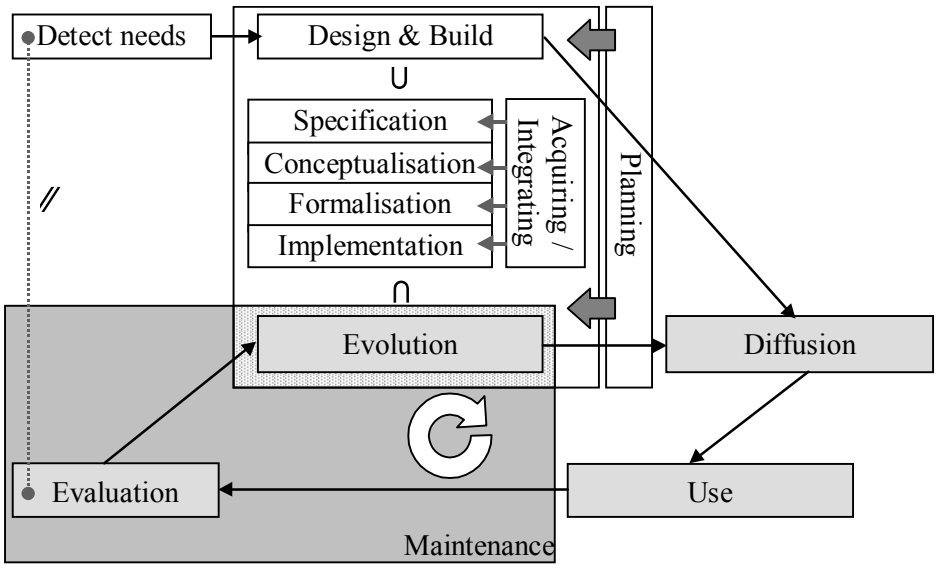


Figure 9 Merged cycles

Even if a full investigation of the complete lifecycle is out of the scope of this document, it must be noticed that ontology maintenance has consequences beyond the ontology lifecycle. It impacts everything that was built using this ontology. A software where the ontology was hardwired has to be versioned, knowledge bases coherence has to be maintained, etc. Therefore, although the problem of the ontology evolution itself is already a complex one, one should consider the fact that ontologies provide building blocks for modelling and implementation. What happens to the elements that were built thanks to these building blocks when a change occurs in the ontology? (e.g. Add a concept, Delete a concept, Modify a concept, Modify hierarchy, Add a relation, Delete a relation, Modify a relation, etc.).

The activities described in the rest of this chapter participate to design and build activities because these are where the major part of the research work is situated.

2.2 Ontology: the engineering

The art of ranking things in genera and species is of no small importance
and very much assists our judgement as well as our memory.
This helps one not merely to retain things, but also to find them.
And those who have laid out all sorts of notions under certain headings
or categories have done something very useful
— Gottfried Wilhelm von Leibniz

[Mizoguchi *et al.*, 1997] explained in one sentence what is the challenge the ontology engineering field must face: "Most of the conventional software is built with an implicit conceptualisation. The new generation of AI systems should be built based on a conceptualisation represented explicitly". In fact, by making at least some aspects of our conceptualisations explicit to the systems, we can improve their behaviour through inferences exploiting this explicit partial conceptualisation of our reality. "Ontology in philosophy contributes to understanding of the existence. While it is acceptable as science, its contribution to engineering is not enough, it is not for ontology engineering which has to demonstrate the practical utility of ontology. It is true that every software has an ontology in itself and every president of a company has his/her own ontology of enterprise. But, such an ontology is implicit. An explicit representation of ontology is critical to our purpose of making computers 'intelligent'. (...) the ultimate purpose of ontology engineering is: 'To provide a basis of building models of all things, in which information science is interested, in the world'." [Mizoguchi *et al.*, 1997]

Since the scientific discipline of Ontology is evolving towards an engineering discipline, it does need principled methodologies [Guarino and Welty, 2000]. In the following part I shall investigate the activities involved in designing an ontology and participating in the ontology lifecycle. Several guidelines and methods have been proposed to design ontologies. I shall try to give an overview of different options proposed so far and I shall try to conciliate the different contributions when it is possible.

2.2.1 Scope and Granularity: the use of scenarios for specification

To see a world in a grain of sand
And a heaven in a wild flower,
Hold infinity in the palm of your hand
And eternity in an hour.
— William Blake

One should not start the development of one's ontology without knowing its purpose and scope [Fernandez *et al.*, 1997]. In order to identify these goals and the limits, one has to clearly state why the ontology is being built, what its intended uses are and who are the stakeholders [Uschold and Gruninger, 1996]. Then, one should use the answers to write a requirements specification document.

Specification will give the scope and granularity of the ontology: a notion has to be included or detailed if and only if its inclusion or details answers a specified need. Inspired by [Charlet *et al.*, 2000], here is an example of variation in granularity around the concepts of *man* and *woman*:

- `man < human / woman < human` : we know there exists two different concepts *man* and *woman* that are descended from the concept *human*.
- `man := human - characteristic - male / woman := human - characteristic - female`: the two different concepts *man* and *woman* are still descended from the concept *human* but now they are known to be different because of a characteristic being *male* or *female*.

- `man := human - attribute - sex - value - male` / `human - attribute - sex - value - female`: the two different concepts *man* and *woman* are still descended from the concept *human* with a differentiating characteristic being *male* or *female*, but we now know that this characteristic is the *value* of the *sex attribute*.
- and so on.

This example is in the context of a medical application and depending on the specifications of the requirements, a level of granularity may be insufficient (hampering the abilities of the system) or useless (costing resources for no reason).

Adapting the characteristics of indexes given in information retrieval [Korfhage, 1997] and the notion of granularity illustrated above, I retain three characteristics of the scope of an ontology:
- *exhaustivity*: breadth of coverage of the ontology, *i.e.,* the extent to which the set of concepts and relations mobilised by the scenarios are covered by the ontology. Beware, a shallow ontology (e.g. one concept 'entity' and one relation 'in relation with') can be exhaustive.
- *specificity*: depth of coverage of the ontology *i.e.,* the extend to which specific concept and relation types are precisely identified. The example given for exhaustivity had a very low specificity; an ontology containing exactly 'german shepherd', 'poodle' and 'labrador' may be very specific, but if the scenario concerns all dogs then its exhaustivity is very poor.
- *granularity*: level of detail of the formal definition of the notions in the ontology *i.e.,* the extend to which concept and relation types are precisely defined with formal primitives. An ontology relying only on subsumption hierarchies has a very low granularity while an ontology in which the notions systematically have a detailed formal definition based on the other ontological primitives, has a very high granularity (c.f. previous example in the context of a medical application).

Three characteristics of the scope of an ontology are its *exhaustivity*, *specificity* and *granularity*. *Exhaustivity* is the breadth of coverage of the ontology *i.e.*, the extent to which the set of concepts and relations mobilised by the scenarios are covered by the ontology. *Specificity* is the depth of coverage of the ontology *i.e.*, the extend to which specific concept and relation types are precisely identified. *Granularity* is the level of detail of the formal definition of the notions in the ontology *i.e.*, the extend to which concept and relation types are precisely defined with formal primitives.

An interesting technique to capture the application requirements in context, is the one of scenario analysis as presented for example in [Caroll, 1997] and used for software engineering. Scenarios are used as the entrance point in the project, they are usually information-rich stories capturing problems and wishes. [Uschold and Gruninger, 1996] uses the notion of motivating scenarios: "The development of ontologies is motivated by scenarios that arise in the applications (...). The motivating scenarios are story problems or examples which are not adequately addressed by existing ontologies. A motivating scenario also provides a set of intuitively possible solutions to the scenario problems. These solutions provide an informal intended semantics for the objects and relations that will later be included in the ontology. Any proposal for a new ontology or extension to an ontology should describe one or more motivating scenarios, and the set of intended solutions to the problems presented in the scenarios. (…) By providing a scenario, we can understand the motivation for the prior ontology in terms of its applications."

[Caroll, 1997] proposes to base the system design activity upon scenario descriptions. Scenarios are a relevant medium for representing, analysing and planning how a system might impact its stakeholders' activities and experiences.

> "The defining property of a scenario is that it projects a concrete narrative description of activity that the user engages in when performing a specific task, a description sufficiently detailed so that design implications can be inferred and reasoned about. Using scenarios in system development helps keep the future use of the envisioned system in view as the system is designed and implemented; it makes use concrete (which makes it easier to discuss use and design use). (...) Scenarios seek to be concrete; they focus on describing particular instances of use, and on user's view of what happens, how it happens, and why. Scenarios are grounded in the work activities of prospective users; the work users do drives the development of the system intended to augment this work. Thus scenarios are often open-ended and fragmentary; they help developers and users pose new questions, question new answers, open up possibilities. It is not a problem if one scenario encompasses, extends, or depends upon another; such relations may reveal important aspects of use. They can be informal and rough, since users as well as developers may create and use them, they should be as colloquial, and as accessible as possible. They help developers and their users envision the outcomes of design -an integrated description of what the system will do and how it will do it- and thereby better manage and control these outcomes." [Caroll, 1997]

Scenarios have the advantage to enable communication in natural language while capturing situation, context, stakeholders, problems and solutions with their associated vocabulary. [Uschold and Gruninger, 1996] exploit the scenarios to introduce informal competency questions. These are queries arising in the scenarios and placing expressiveness requirements on the envisioned ontology: the ontology must be able to represent the informal competency questions and characterise their answers. "The competency questions specify the requirements for an ontology and as such are the mechanism for characterising the ontology design search space. The questions serve as constraints on what the ontology can be, rather than determining a particular design with its corresponding ontological commitments. There is no single ontology associated with a set of competency questions." [Uschold and Gruninger, 1996]

Scenario analysis, as many other activities in ontology engineering, is not a one-off activity, but will be pursued during the whole ontology design process and lifecycle. New scenarios will arise, existing scenarios will be refined. [Fernandez *et al.*, 1997] also note that the inspection of glossary terms (lexicons) without looking into the details of the definitions can help at that stage to define the scope of the ontology. We shall see that this is closely related to the middle-out perspective of ontology engineering.

2.2.2 Knowledge Acquisition

Figure 10 Caricature of the knowledge engineer

Data collection or knowledge acquisition is a collection-analysis cycle where the result of a required collection is analysed and this analysis triggers new collections. Elicitation techniques help elicit knowledge. Several techniques exist for data collection and benefit from two decades of work in the knowledge acquisition community (see, for example, [Dieng, 1990], [Dieng, 1993] and [Dieng *et al.*, 1998] as well as [Sebillotte, 1991], [La France, 1992] and [Aussenac, 1989]). Experts, books, handbooks, figures, tables and even other ontologies are sources of knowledge from which the knowledge can be elicited using in conjunction techniques such as: brainstorming, interviews, formal and informal analysis of texts, and knowledge acquisition tools [Fernandez *et al*., 1997]

The elicitation techniques are typically associated with bottom-up approaches. They may also be used in a top-down approach guided by models such as the ones of CommonKADS [Breuker and Van de Velde, 1994], to elicit data required by the models. In fact, as noticed in [Fernandez *et al.*, 1997], they influence the whole process of engineering and maintenance, and different techniques may be useful at different stages of the process. A given source can be exploited in one or more of the perspectives adopted for ontology engineering depending on the nature of the source. For example interviews are prone to bottom-up and middle-out perspectives, whereas integrating ontologies leads to top-down and middle-out perspectives.

[Uschold and Gruninger, 1996] used brainstorming sessions to produce all potentially relevant terms and phrases; at this stage the terms alone represent the concepts, thus concealing significant ambiguities and differences of opinion.

[Fernandez *et al.*, 1997] used:
- *Non-structured interviews*: with experts, to build a preliminary draft of the requirements specification document.
- *Informal text analysis*: to study the main concepts given in books and handbooks. This study enables you to fill in the set of intermediate representations of the conceptualisation.
- *Formal text analysis*: The first thing to do is to identify the structures to be detected (definition, affirmation, etc.) and the kind of knowledge contributed by each one (concepts, attributes, values, and relationships).
- *Structured interviews*: with experts to get specific and detailed knowledge about concepts, their properties and their relationships, to evaluate the conceptual model once the conceptualisation activity has been finished, and to evaluate implementation.

It should be noticed that among the documents studied during data collection, there can be existing terminologies or even ontologies collected from the domain. This will then lead to the process of integrating other ontologies in the ontology being built.

The data collection is not only a source of raw material: for example interviews to expert might help to build concept classification trees and to contrast them against figures given in books [Fernandez *et al.*, 1997].

Knowledge acquisition and modelling is a dialog and a joint construction work with the stakeholders (users, sleeping partners, managers, providers, administrators, customers, etc.). They must be involved in the process of ontology engineering, and for these reasons semi-formal/natural language views (scenarios, tables, lists, informal figures) of the ontology must be available at any stage of the ontology lifecycle to enable interaction between and with the stakeholders.

Data collection is a goal-driven process. People in charge of the data collection always have an idea of what they are looking for and what they want to do with the collected data. It is essential to consider the end product that one desires right from the start (scenarios, models, ontologies…) and from that to derive what information should be identified and extracted during the data collection. The models will also provide facets, views or points of view that are useful to manage and modularise the collection products: organisational view, document view...

Interviews: Interviews can be individual or grouped, they can take place with a large or small number of people, and they can be one-off interviews or 'repeater interviews'. Depending on these circumstances the techniques and the organisation of the interviews can vary a lot. Principally an interview can be extremely structured (interrogation) or completely free (spontaneous expression), but the most common one is the semi-structured interview which can be anywhere on the continuum between these two extremes. A classic plan of progress during a semi-structured interview is:

1. Opening discussion: This first part is typically unstructured. To initiate the dialog first questions have to be very general, very broad. A good subject to start with is the tasks and roles of the people interviewed. Interviewee are encourage to speak until a long silence sets up. If need be, spontaneous expression can be kept running using the journalists' short questions (why? how? when? who? where? with what? any thing else?).
2. Flashback and clarification: Once reached a terminal silence ("-anything else? … -Hum no!") a more structured part begins. It implies to take notes during the first part and to build some informal personal representation so as to be able to identify points and subjects to be clarified or detailed through flashback questions. It is also in this part that questions that have been prepared before according to the information that is looked for and that has not been spontaneously answered in the first part can be asked (e.g. strategic aspects detected in the scenarios).
3. Self-synthesis: Finally, a good idea is to make interviewed people synthesise, summarise or analyse themselves what they said during the interview and make them conclude.

As the collection goes on, interviews tend to be more and more structured since the collection tend to be more and more focused.

Observations: As for interviews, observation comes with a broad range of options. It can be anywhere on the continuum between spying someone in his everyday activity without him knowing it (not very ethical, but extremely pure and natural view) and asking someone to simulate his activity in front of you in a reconstruction fashion (much more acceptable, but it may skew or distort some real issues). The observation can be about people, the way they work (with or without making them comment), on a real task or on simulated scenario, in real time, recorded or based upon traces of the activity. It can also be focused on very specific aspects (documents manipulated, desk organisation, acquaintance network, etc.). Depending on the actor the interesting situation may be very different.

Document Analysis: [Aussenac-Gilles *et al.*, 2000] explain that documents or any natural language support (messages, text files, paper books, technical manuals, notes, protocol transcripts, etc.) are one of the possible forms the knowledge may take. Authors promote a new approach for knowledge modelling based on knowledge elicitation from technical documents which benefits from the increasing amount of available electronic texts and of the maturity of natural language processing tools. The acquisition of knowledge from texts applies natural language processing tools and based on results in linguistics to semi-automate systematic text analysis and ease the modelling process.

[Aussenac-Gilles *et al.*, 2000] distinguish and describe four major steps in this process:

1. **Setting up the corpus:** From the requirements that explain the objectives underlying the model development, the designer selects texts among the available technical documentation. He must be an expert about texts in this domain to characterise their type and their content. The corpus has to cover the entire domain specified by the application. A glossary, if it exists, is useful to determine sub-domains and to verify that they are well covered. The corpus is then digitalised if it was not. Beginning the modelling may lead to reconsider the corpus.
2. **Linguistic study:** This step consists in selecting adequate linguistic tools and techniques and in applying them to the text. Their results are sifted and a first linguistic based elicitation is made. The objective is to allow the selection of the terms and lexical relations that will be modelled. The results of this stage are quite raw and will be further refined.
3. **Normalisation:** This step includes two parts. The first part is still linguistic, it refines the previous lexical results. The second part concerns the semantic interpretation to structure concepts and semantic relations. The modelling goes from terminological analysis to conceptual analysis, that means from terms to concepts and from lexical relations to semantic ones. During normalisation, the amount of data to be studied is gradually restricted.
4. **Formalisation:** The formalisation step includes building and validating the ontology. Some existing ontologies may help to build the highest levels and to structure it into large sub-domains. Then semantic concepts and relations are translated into formal concepts and roles and inserted in the ontology. This may imply to restructure the ontology or to define additional concepts, so that the inheritance constraints on the subsumption links are correct. Inserting a new concept triggers a local verification to guarantee the syntactic validity of the added description. A global validation of the formal model is performed once the ontology reaches a quite stable state to verify its consistency.

"The modelling work must be carried out from documents attested in the practice of a domain and gathered in a corpus.(...) The selection [of the documents] is based on criteria pertaining to the analysis method used (corpus analysis, for instance) and to the problem to be solved (to keep only the documents relevant for the problem to be solved). Setting up a corpus is delicate: the choice of a corpus introduces bias, that we may not be able to evaluate." [Bachimont, 2000] Here the definition of the scope and especially the scenarios are of valuable help to choose the corpus and the criteria for gathering and analysis.

NLP tools are extremely interesting to scale up the process of data-collection because they can assist and partially relieve the knowledge engineer when large corpora have to be analysed. However the document analysis aspect of data-collection is not limited to the linguistic content of documents. Graphical documents may be very rich too: organisation charts, maps, tables, figures, etc. as shown in Figure 11. For these documents few assisting tools exists and large corpora of drawings, for instance, cannot be analysed by one person in charge of knowledge engineering or management.

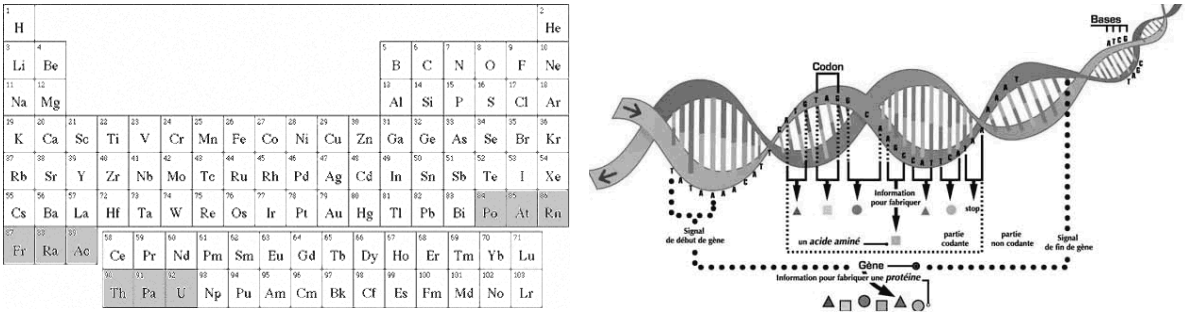


table of chemical elements schema of DNA structure

Figure 11 Examples of documents that cannot be included in a NLP corpus.

Other aspects of documents are interesting such as the effective use of the document (e.g. compare an empty form with a filled form) or their flows in an organisation (what are the typical pathways of a given type of document).

Questionnaire & Questioning: Questionnaires are a relatively inexpensive way of getting people to provide information. However, elaborating a questionnaire is a critical job and a good questionnaire has to be tested and reviewed several times to validate its usability just like any artefact. Note that in many situations, other data collection techniques are superior.

From a general point of view, it may seem trivial, but the questions must be formulated so that each respondent clearly understands the topic, the context and the perspective. The profile and the role of the respondents must be considered when choosing the question so that the persons are not asked to give information that they do not have. The order and the way questions are formulated may influence the answers, so one must be very careful not to skew the collection or to miss something.

Brainstorming & Brainwriting: It was originally a group problem-solving technique. More generally speaking, it consist of a meeting session based on spontaneous and unrestrained contributions from all members of the group. The discussion is centred on a specific theme, a set of ideas or problems with the purpose of generating new ideas or solving the problems.

It can be noticed that if both brainstorming and interviews have to be used, it is better to run the interviews before (at least the first wave) so that some ideas have already been gathered to sustain the discussion and that the exchange of ideas done during brainstorming will not bias the individual interviews.

In some cases, hierarchy or other background influence may hamper a brainstorming session (e.g. someone may not want to publicly go against the opinion of his chief, someone may be too shy to speak). It is therefore extremely important to carefully choose participant. It is also a case where brain-writing can be used: people write down their ideas, put the paper in a basket and they are anonymously written down and discussed. An alternative is also to make people re-draw the papers from the basket, write their comments, put it back and so on until everyone has written something on all the pieces of paper.

New technologies can enable other forms of discussions (e.g. news, mailing-lists…), but here again human contact is usually the best option.

During data collection, several drafts and intermediate versions of data collection reports, structures and traces (e.g. scenario reports, lexicons, interview transcriptions, pictures of observations...) are generated. Analysing each one of them provide guidance and focus for further data collection. There may be several analysis of one version, for instance one may have to analyse a product again after discovering new areas of interest that had not been identified yet when the first analysis occurred. A product can also be analysed by several people. For these reasons, it is important that the product be kept intact with the exact wording, phrasing and representations captured by data collection.

The analysis of a report resulting from a data collection session can be divided in several phases:

1. Recognition: First the report has to be reviewed entirely without trying to structure or link together the information it contains. One must concentrate on identifying the blocks (e.g. a definition or a schema), the elements (e.g. a term denoting a concept) and the connectors (e.g. logical and chronological connectors).
2. Then, the report is reviewed several times not only to make sure every interesting bit has been identified, but also and mainly to structure and bring out the links, relationships, dependencies, grouping and cross references. the review is not a linear reading.
3. Once it seems everything has been extracted, the analysis gives an informal, but structured, annotated and tidy report of what has been collected.

Electronic copies of the reports do facilitate manipulation and versioning.

Based on the analysis the following wave of data collection will probably be more focused trying to confirm or invalidate intermediary results and to gather further details. This can lead to more structured interviews, focused observation, discussions with the people about the intermediary results, or one-off communication (e.g. phone or e-mail) to clarify a special point. Once the informal models are stable enough they represent the starting point for formalisation.

2.2.3 Linguistic study & Semantic commitment

To use the same words is not a sufficient guarantee of understanding; one must use the same words for the same genus of inward experience; ultimately one must have one's experiences in common.
— Friedrich Nietzsche

In this part, we explain the major phases when building an ontology from the data collection analysis reports. The activities and guidelines given here are sometimes applicable to reuse and integration of ontologies; I shall address specific matters of reuse in a latter part. It is important to notice that from this stage of the ontology lifecycle, literature on the subject acknowledges the importance of reaching and maintaining a commitment between stakeholders on the ontological work and the results.

[Bachimont, 2000] decomposes the ontology modelling process in three stages, corresponding to three commitments, the first of which is the semantic commitment specifying the linguistic meaning of notions that were identified during data collection. Notions are knowledge pieces enabling to define other knowledge pieces, but are also defined by these knowledge pieces. The problem is that there does not exist general primitives in a domain while it is necessary for us to have at our disposal primitives to undertake the knowledge modelling. Therefore, defining an ontology is not about characterising or determining already existing primitives in a domain, but it is about modelling or designing primitives for the problem solving. How, do we design these primitives then? The idea is to start again from the linguistic expression of the knowledge of the domain [Bachimont, 2000].

The terminological study is at the heart of ontology engineering, it produces the candidate terms for which consensual definitions have to be produced. One of the key issues is the context of the knowledge: every piece of knowledge is tuned to a context in which it is applied [Mizoguchi *et al.*, 1997]. Summarising [Bachimont, 2000], the texts contain linguistic expressions of the targeted notions and it is tempting to define these notions just by these wordings. The advantage is that the notions receive straightaway the ability to be interpreted by specialists using them. The inconvenience is that, if these wordings are interpretable, nothing imposes that they must be interpreted the same way, or at least in a coherent and compatible way. Therefore, it is necessary to constrain the spontaneous interpretation of the wordings so that, respecting these interpretation constraints, any specialist associates the same meaning as his colleagues to a wording; the problem is to start again from the semantics of natural language to reach the non contextual definition of a wording. Therefore, the first thing to do is to capture and fix the context in which the terminology will be established and shared [Mizoguchi *et al.*, 1997]. Fixing the context is the very first step of the semantic normalisation: among possible meanings for a unit in context, one fixes the one that should always be associated; this amounts to choose a referential context in which, in principle, terms must be interpreted [Bachimont, 2000]. Semantic normalisation is the choice of a reference context, corresponding to the application scenario that motivated knowledge modelling. This point of view enables the modeller to fix what must be the meaning of a wording, *i.e.,* what is the notion behind.

For [Uschold and Gruninger, 1996], an ontology may take a variety of forms, but necessarily it will include a vocabulary of terms and some specifications of their meaning (*i.e.,* definitions). In fact, this lexicon is the main result of a terminological study of the data collection reports; thus the terminological study is usually the next stage of the semantic commitment after fixing the context. As we see now, the major part of the analysed resources is textual, be it documents directly collected or reports or traces resulting from other data-collection techniques. This plainly justifies the interest of tools facilitating textual document analysis.

During scope definition and data collection, some terms and most probably some central terms will already have been identified. In [Uschold and Gruninger, 1996], for example, the authors reuse the competency questions to start their informal terminology by extracting the set of terms used in expressing the question; in addition they produce informal definitions of the terms. "The informal

dictionaries and glossaries defined using this methodology provide the intended semantics of the terminology and lay the foundations for the specification of axioms in the formal language." [Uschold and Gruninger, 1996].

The identification of linguistic expressions in the textual resources, aims to include and define these wordings and terms in lexicons to prepare modelling. The expressions are included and defined if and only if they are relevant to the progress of the application scenarios that are being considered.

Linguistic expressions that were included, or noted as borderline with reference to the previously agreed requirements document are to be defined in the lexicon. [Uschold and Gruninger, 1996] give guidelines to generate the definitions:
- Produce a *natural language text definition*, being as precise as possible.
- Ensure *consistency* with terms already in use; in particular:
- Make ample *use of dictionaries*, thesauri and other technical glossaries.
- Avoid to introduce new terms where possible.
- Indicate the *relationship* with other commonly used terms that are similar to the one being defined (e.g. synonyms or variants with same underlying notion, but perhaps from different perspectives).
- *Avoid circularity* in defining terms; this increases clarity in general and is essential for later formalisation.
- The *definition of each term is intended to be necessary and sufficient* as far as this is possible in natural language. Provide clarification or additional information essential to understanding the definition as separate notes following the definition.

It is also noticed in several papers that one should give examples where appropriate, to clarify distinction or subtleties of definitions. Compared to definition in intension or in extension, examples can be defined as a representative sample of the extension: a good set of examples limited in size and covering as much as possible the different notions specialising the considered one. For instance, examples of 'fruits' should not be chosen only from the 'red fruits' or the example may badly bias the interpretation of the definition; the characteristics should be varied as much as possible: bananas, apples, grapes, lemons, coconuts and apricots are examples of fruits.

As we said before, throughout the ontology design process, it is vital to reach and maintain an agreement with the scenario stakeholders. [Uschold and Gruninger, 1996] experienced considerable variation in the degree of effort required to agree on definitions and terms for underlying concepts: "For some terms, consensus on the definition of a single concept was fairly easy. In other cases several terms seemed to correspond with one concept definition. In particular, there were several cases where commonly used terms had significantly different informal usage, but no useful different definitions could be agreed. This was recorded in notes against the definition. Finally, some highly ambiguous terms are identified as corresponding with several closely related, but different concepts. In this situation, the term itself gets in the way of a shared understanding."

Thus, there exist three cases when studying terms:
- *One term* corresponding to one and only *one definition*: there is no problem.
- *Several terms* corresponding to *one definition*: The terms are *synonyms*, one should keep the list of synonyms and choose one preferred term to refer to that definition.
- *One term* corresponding *several concepts*: This corresponds to an ambiguity: *homographs*.

When faced with an ambiguous term, [Uschold and Gruninger, 1996] suspend use of the term; they give the different notions meaningless labels such as x_1, x_2, x_3 etc. so they can be conveniently referred to in a neutral way; they clarify the notions by carefully defining them using as few technical terms as possible, or only those whose meaning is agreed (consult the dictionary, thesauri, and/or other technical glossaries); they determine which, if any, of the concepts are important enough to be in the ontology; finally, they choose a wording for these notions ideally avoiding the original ambiguous ones.

Being a commitment, the normalisation is a joint work between the knowledge engineer and the stakeholders.

Labelling concepts with a wording is both convenient and dangerous. It is a major source of 'ambiguity relapse' where people happily relapse in ambiguity using the wording according to the definition they associate with it and not the definition actually associated to it in the process of semantic commitment. As proposed above, an interesting exercise when ambiguity is lurking around is to replace labelling wordings by meaningless identifiers; this obliges the designers to always refer to the definition and thus make sure it is the right intension. However it is obvious that it can only be used for modelling purposes and not for interfaces with the end-users. Presentation, re-presentation, views and interfaces are needed to bridge the gap between conceptual structures and user concerns.

2.2.4 Conceptualisation and Ontological commitment

The second stage identified in [Bachimont, 2000] is the ontological commitment specifying the formal meaning of the notions. This passage from linguistic wording to the formal notions is bridged by the rigorous lexicons that were obtained in the previous stage.

During the terminological study, some terms and definitions are naturally found to be close because they belong to the same theme or subject, the same interest area, the same domain, the same activity field, etc. This grouping is to be encouraged and then actively performed. [Uschold and Gruninger, 1996] structure the terms loosely into work areas corresponding to naturally arising sub-groups; groups arose such that terms are more related to other terms within the same group than they are to terms in other groups. This could be compared to clustering based on closest neighbours. They group similar terms and potential synonyms together into areas and then they identify semantic cross-references between the areas *i.e.*, concepts that are likely to refer to or to be referred to by concepts in other areas. Authors suggest that this information can be used to help identify which work area to tackle first to minimise likelihood of rework. They address each work area in turn, starting with work areas that have the most semantic overlap with other work areas. These are the most important to get right in the first place, because mistakes lead to more re-work. [Uschold and Gruninger, 1996]

Conceptualising leads ontology designers to organise the notions and to do so they look for relations between these notions that could be used to structure them. The first one we have seen is close to the lexical fields or semantic similarity. The second is a partition, indeed, from the definitions, it rapidly emerges that some terms have different roles: some terms denote natural concepts and others denote relations between these concepts. The relations have their definition in which it is important to include the types of concepts they link.

Relations can be labelled with verbs, or verbal groups, thus one criteria for partitioning linguistic expressions can be to separate verbs and non verbs; it is the case in the intermediary representations of [Fernandez *et al.*, 1997].

As explained in the following extract, the conceptualisation process is a refinement activity, iteratively producing new refined representations of the notions being modeled (intermediary representation as they are called in [Gómez *et al.*, 1996]). As we shall see, this is the beginning of a continuum between informal textual definitions of the notions at play in a scenario, and implemented representation of these notions in formal languages. Every new intermediary representation is a step toward the required degree of formalisation.

" The first thing to do is to build a complete Glossary of Terms (GT).(...) Once you have almost completed the complete Glossary of Terms, you must group terms as concepts and verbs. Each set of concepts/verbs would include concepts/verbs that are closely related to other concepts/verbs inside the same group as opposed to other groups. Indeed, for each set of related concepts and related verbs, a concepts classification tree and a verbs diagram is built. After they have been built, you can split your ontology development process into different, but related, teams.

Concepts will be described using: Data Dictionary, which describes and gathers all the useful and potentially usable domain concepts, their meanings, attributes, instances, etc.; tables of instance attributes, which provide information about the attribute or about its values at the instance; tables of class attributes, to describe the concept itself, not its instances; tables of constants, used to specify information related to the domain of knowledge that always take the same value; tables of instances, which define instances; and attributes classification trees, to graphically display attributes and constants related in the inference sequence of the root attributes, as well as the sequence of formulas or rules to be executed to infer such attributes.

Verbs represent actions in the domain. They could be described using (...): a Verbs Dictionary, to express the meaning of verbs in a declarative way; tables of conditions, which specify a set of conditions to be satisfied before executing an action, or a set of conditions to be guaranteed after the execution of an action. Finally, to say that tables of formulas and tables of rules gather knowledge about formulas and rules. Note that both branches use these two intermediate representations." [Fernandez *et al.*, 1997]

2.2.5 Taxonomic skeleton

As we have seen with the set of intermediary representations proposed above, ontology usually includes a taxonomy of concepts. "One of the principal roles of taxonomies is to impart structure on an ontology, to facilitate human understanding, and to enable integration." [Guarino and Welty, 2000]. The definition of the notions gives their intensions that defined and organised in the taxonomy. The basis structural principle of the differential definition of intensions says that the defined notion is situated in relation to a genus according to a differentia. The notions of genus and differentia go back to Aristotle [Sowa, 2000b] who defined species by giving its genus (genos) and its differentia (diaphora). The genus is the kind under which the species falls. The differentia is what characterises the species within that genus. Applying this principle, [Kassel *et al.*, 2000] gives the example of a definition of a 'message' based on the genus 'document': "*a message is a document addressed by a sender to an addressee*"; the differentia is characterised here by the type of role played by the document in communication.

As stated in the definitions, a taxonomy is a classification based on similarities. It is not surprising to find this structure at the heart of knowledge modelling since, as I showed before, it is the counterpart of three of the most elementary inferences we use everyday almost as a reflex:

- *classification or identification*: the inference whereby we determine if something belongs to a class or a category e.g. if you see a horse, if you recognise it as a horse, it means you have been able to classify/identify the object we were seeing as belonging to the class/category of concepts labelled "horse".
- *categorisation*: the inference whereby we identify and organise categories/classes of things in our world. If you have green peas, green beans, tomatoes and strawberries on a table and you are asked to divide them into two categories based on one criterion, you will most probably and quite easily create a "green things" category and a "red things" category.
- *generalisation/specialisation*: the inference whereby we jump from a notion to a more general/precise one.

The relation at the heart of the taxonomy is the subsumption: a concept C_1 subsumes another concept C_2 if and only if all instances of C_2 are necessarily instances of C_1; *"to subsume"* means to incorporate a concept under a more general one. Based on the subsumption link, is defined the

inheritance mechanism, whereby a concept inherits from the characteristics of a concept that subsumes it. The nature of the structure is still a debate, especially between people believing in a pure tree structure, people in favour of a lattice structure, or people in favour of a general multi-inheritance graph. We shall present here seminal contributions to the rationalisation of the global structuring of the taxonomy.

To build the taxonomy of concepts, several approaches have been opposed in literature:

- *Bottom-Up approach*: starting from the most specific concepts and building the structure by generalisation; the ontology is built by determining first the low taxonomic level concepts and by generalising them. This approach is prone to provide tailored and specific ontologies with fine detail grain concepts.
- *Top-Down approach*: Starting from the most generic concept and building a structure by specialisation; the ontology is built by determining first the top concepts and by specialising them. This approach is prone to the reuse of ontologies and inclusion of high level philosophical considerations which can be very interesting for coherence maintenance.
- *Middle-Out approach*: Identifying central concepts in each area/domain identified; core concepts are identified and then generalised and specialised to complete the ontology. This approach is prone to encourage emergence of thematic fields and to enhance modularity and stability of the result.

The choice of an approach and its motivations are closely linked to the domain of intervention and the type of data manipulated. For example, [Uschold and Gruninger, 1996] argues that middle-out approach is the best approach. First they believe that the bottom-up approach results in a very high level of detail which increases overall efforts, makes it difficult to spot commonality between related concepts and increases risk of inconsistencies leading to re-work. Secondly they think that, starting at the top, a top-down approach results in choosing and imposing arbitrary high-level categories leading to less stable models which in turn lead to re-work. "A middle-out approach, by contrast, strikes a balance in terms of the level detail. Detail arises only as necessary, by specialising the basic concepts, so some effort is avoided. By starting with the most important concepts first, and defining higher level concepts in terms of these, the higher level categories naturally arise and thus are more likely to be stable. This, in turn, leads to less re-work and less overall effort." [Uschold and Gruninger, 1996] Middle-out approach is linked to the idea of identifying relevant thematic groups of terms. However, as noticed by the authors themselves, the notion of basic concept can be tightly dependent on the application context. "For example, 'dog' is basic, 'mammal' is a generalisation, and 'cocker spaniel' is a specialisation. While differences arise for individuals of widely varying expertise in an area, (e.g. a dog breeder may regard a particular species as basic), broadly, what is basic is the same for most people." [Uschold and Gruninger, 1996]

2.2.5.1 Differential semantics [Bachimont, 2000]

Following the linguistic study presented in the previous part, [Bachimont, 2000] states that the global structuring of the ontology (the taxonomy) is a tree:

> The differential semantics enables to describe units between themselves by the similarities which unite them and the differences that distinguish them. Now, described units have in common the ability to be comparable: they have in common the characteristic that 'they say or they are something' so that in a second stage we can determine that they are not the same thing. Therefore, each unit determines itself from one ultimate generic unit, a root unit, to which it belongs. [For instance a common root for ontologies is the "Thing" concept.]
>
> Moreover, all units determine themselves on the one hand from the generic unit they belong to, on the other hand from the differences that distinguish them. This means that the network of units is a property inheritance network where children units inherit sememes from a generic parent unit.
>
> Therefore, the problem is to determine the structure of such an inheritance network. (...) Suppose that a unit U has several direct parent units, let's say two. These two parent units are distinct, they determine themselves from similarities and differences between themselves. If they are characterised only by similarities between themselves, it implies that one is generic compared to the other, and therefore, only the other one is a direct parent. Therefore, they must be characterised by some differences that distinguish them. This implies that they determine themselves from mutually incompatible features (...) Therefore, a given unit can only have one and only one parent unit. So the network must be a tree. [Bachimont, 2000]

[Bachimont, 2000] proposes to determine the meaning of a unit in the tree, a node, using four differential principles. When applied to a given node, these principles make explicit similarities and differences with its neighbours. These principles are:

- The *parent community principle*: any unit is determined by the similarities it has in common with its parent; one must make explicit its similarities to the parent unit. It is, mutatis mutandis, the Aristotelian principle of definition by close kind (genos). This principle is at the core of the generalisation process.
- The *parent difference principle*: any unit is different from its parent, otherwise there would be no point to define it; one must make explicit the difference that distinguishes it from its parent unit. It is, mutatis mutandis, the Aristotelian principle of definition by specific difference (diaphora). This principle is at the core of the specialisation process.
- The *brother difference principle*: any unit is different from its brothers, otherwise there would be no point to define it; one must make explicit the difference that distinguishes it from its brother units. This principle is not Aristotelian, but it comes from the differential paradigm. The differences are the specificities of the differentia for the different brothers.
- The *brother community principle*: by definition all the children units of a parent unit have in common the same generic feature as the one they share with their parent unit. But one must establish another community between children units: the one that enables us to define the differences mutually exclusive between children units.

Bachimont gave the example of a function of a person in a movie: three concepts can be organised according to the previous definitions: Function, Actor, Director. These principles are:

- The parent community principle: Actor and Director are Functions.
- The parent difference principle: Actor and Director are restrictions of Function for a person involved a movie.
- The brother difference principle: an Actor plays in the film, a Director makes the film.
- The brother community principle: Both Actor and Director are Functions of a person in a film.

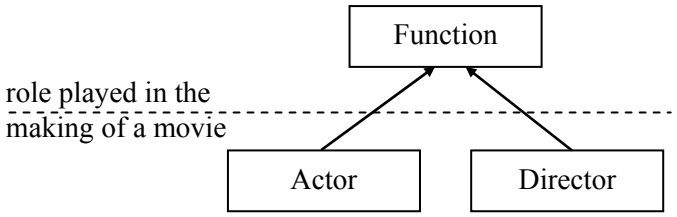


Figure 12 Example of Bachimont

An assisting design tool is currently being developed and tested [Troncy and Isaac, 2002] for this approach of the structuring.

2.2.5.2 Semantic axis [Kassel *et al.*, 2000]

One point where I do not entirely agree with Bachimont, is the necessity of defining only one type of difference with the parent unit. I can quite easily find cases where one unit can be specialised in several ways. For instance the *genos* 'packaging' could be specialised along several types of differences: "recyclable / non recyclable", "shock-proof / fragile", etc. "By considering the definitions we can bring out similarities between the differentia. (...) These groupings correspond to the notion of semantic axis" [Kassel *et al.*, 2000]. A semantic axis groups the children of a concept according to the characteristics involved in the definition of their differentia. This notion reifies the brother community and difference principles. For instance: documents distinguished by their medium (paper, digital...), documents distinguished by their role (message, reference...), document distinguished by their status (official, informal, etc.), and so on.

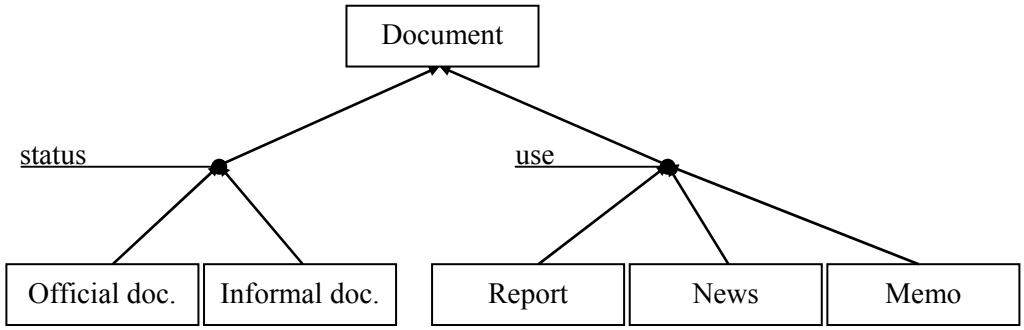


Figure 13 Example of Kassel and colleagues.

These semantic axis introduce points of views in the inheritance hierarchy. They do not oblige the designer to choose one and only one criteria of differentia below a genus. This is close to viewpoints [Ribière, 1999] or facets in object-oriented modelling languages where different taxonomies are built and 'footbridges' are sets to indicate equivalence between two concepts of two different taxonomies. It is also close to the problematic of reconciling two ontologies.

Building upon this work Gilles Kassel proposed OntoSpec [Kassel, 2002] a method for the semi-informal specification of ontologies. It reuses results from the work of Guarino and Welty presented thereafter and builds a strong experience and survey of the field to propose a taxonomy of primitives, and a set of design rules to specify and structure the ontology. The development and use of tools supporting this work will be an interesting experience to follow.

2.2.5.3 Checking the taxonomy [Guarino and Welty, 2000]

Guarino and Welty also made a significant contribution to the theoretical foundations of the field. In 1992, Guarino started by distinguishing natural concept, role, attributes, slots, qualities. He proposed to use the term 'role' only in Sowa's sense; bearing on Husserl's theory of foundation, he distinguishes between roles and natural concepts, and defines a role as a concept which implies some particular 'pattern of relationships', but does not necessarily act as a conceptual component of something. He defines 'attributes' as concepts having an associate relational interpretation, allowing them to act as conceptual components as well as concepts on their own; He proposes a formal semantics which binds these concepts to their corresponding relations, and a linguistic criterion to distinguish attributes from 'slots', *i.e.,* from those relations which cannot be considered as conceptual components. Moreover, he shows how the choice of considering attributes as concepts enforces 'discipline' in conceptual analysis as well as 'uniformity' in knowledge representation. [Guarino, 1992]

This first work led him to the following definitions, where the nec operator is defined as *the modal necessity (box) operator,* $\leq$ is the part of relation and $/\leq$ is its negation.

Definition: The concept α is founded on β (written $\alpha \Downarrow \beta$) if: **nec** $\forall x \ (x \in \alpha \supset \exists y \ (y \in \beta \wedge x /\leq y \wedge y /\leq x))$.

Definition: α is called founded (written $\alpha \Downarrow$) if there exists a β such that $\alpha \Downarrow \beta$. α is called essentially independent (written $I(\alpha)$) if $\neg (\alpha \Downarrow)$, and self-founding if $\alpha \Downarrow \alpha$.

Definition: A concept α is called semantically rigid (written $R(\alpha)$) if $\forall x \ (x \in \alpha \supset$ **nec** $(x \in \alpha))$.

Definition: A concept α is called a role if it is founded, but not semantically rigid, that is, $\alpha \Downarrow \wedge \neg R(\alpha)$.

Definition: A concept α is called a natural concept if it is essentially independent and semantically rigid, that is, $I(\alpha) \wedge R(\alpha)$.

In [Guarino and Welty, 2000] is presented an additional and exemplified comprehensive set of definitions aiming at providing the ontologists with methodological elements. The first notion presented is the notion of identity: "Strictly speaking, identity is related to the problem of distinguishing a specific instance of a certain class from other instances by means of a characteristic property, which is unique for it (that whole instance)." The identity relies on the existence of a rigid notion:. "A rigid property is a property that necessarily holds for all its instances." [Guarino and Welty, 2000]. Authors give the example of *being a Person* which is rigid since an instance *x being a Person* cannot cease to be a Person unless the instance ceases to be. On the other hand, *being a Student* is anti-rigid since all the instances of Student have the ability to cease to be a Student without ceasing to be (that is a chance for me !). Authors introduces the following notations:

Rigid	ϕ^{+R}	*ϕ is a necessary property for all its instances*
Non-Rigid	$\phi^{\sim R}$	*ϕ is not a necessary property for all its instances*
Anti-Rigid	ϕ^{-R}	*ϕ is an optional property for all its instances*

Anti-rigidity attempts to capture the intuition that all instances of certain properties must possibly not be instances of that property.

Definition: An identity condition (IC) for a property ϕ is defined as a relation ρ satisfying $\phi(x) \wedge \phi(y) \rightarrow (\rho(x,y) \leftrightarrow x = y)$.

The authors distinguish between supplying an IC and simply carrying an IC: non-rigid properties can only carry their ICs, inheriting those supplied by their subsuming rigid properties.

Definition: Let ϕ be a rigid property, and $\Gamma(x,y,t,t')$ a formula containing x, y, t, t' as the only free variables, such that $\neg\forall xytt'(\Gamma(x,y,t,t') \leftrightarrow x = y)$. We say that ϕ carries the IC Γ *iff* one of the two following definitions is verified.

Definition: Γ is a necessary IC carried by ϕ when

$$E(x,t) \wedge \phi(x,t) \wedge E(y,t') \wedge \phi(y,t') \wedge x=y \rightarrow \Gamma(x,y,t,t')$$
$$\neg\forall xy(E(x,t) \wedge \phi(x,t) \wedge E(y,t') \wedge \phi(y,t') \rightarrow \Gamma(x,y,t,t'))$$

Definition: Γ is a sufficient IC carried by ϕ when

$$E(x,t) \wedge \phi(x,t) \wedge E(y,t') \wedge \phi(y,t') \wedge \Gamma(x,y,t,t') \rightarrow x=y$$
$$\exists xytt'\ \Gamma(x,y,t,t')$$

ICs are 'inherited' along a hierarchy of properties.

Definition: A non-rigid property carries an IC Γ *iff* it is subsumed by a rigid property carrying Γ.

Definition: Any property carrying an IC is marked with the metaproperty +I (-I otherwise).

Definition: A property ϕ supplies an IC Γ *iff* i) it is rigid; ii) it carries Γ; and iii) Γ is not carried by all the properties subsuming ϕ. This means that, if ϕ inherits different (but compatible) ICs from multiple properties, it still counts as supplying an IC.

Definition: Any property supplying an IC is marked with the metaproperty +O (-O otherwise). The letter "O" is a mnemonic for "own identity".

From the above definitions, it is obvious that +O implies +I and +R.

Definition: Any property carrying an IC (+I) is called a sortal.

We can summarise the notation as follows:

Carrying an IC	ϕ^{+I}	ϕ carries an identity condition
Not carrying an IC	ϕ^{-I}	ϕ does not carry an identity condition
Supplying an IC	ϕ^{+O}	ϕ supplies an identity condition
Not supplying an IC	ϕ^{-O}	ϕ does not supply an identity condition

The second important notion discussed in [Guarino and Welty, 2000] is Unity in order to distinguish the parts of an instance from the rest of the world by means of a unifying relation that binds them together without involving anything else.

Definition: Let ω be an equivalence relation. At a given time t, an object x is a contingent whole under ω if $\forall y(P(y,x,t) \rightarrow \forall z(P(z,x,t) \leftrightarrow \omega(z, y,t)))$

Definition: Let ω be an equivalence relation. An object x is an intrinsic whole under ω if, at any time where x exists, it is a contingent whole under ω.

If an object is always atomic (*i.e.*, it has no proper parts), then it is an intrinsic whole under the identity relation.

Definition: A property ϕ carries a unity condition (+U) iff there exists an equivalence relation ω such that all its instances are intrinsic wholes under ω.

As an example, authors distinguish three main kinds of unity for concrete entities (*i.e.*, those having a spatio-temporal location):

- *Topological unity*: based on some kind of topological connection (a piece of coal, a lump of coal)
- *Morphological unity*: based on shape (a ball, a constellation)
- *Functional unity* (a hammer, a bikini)

As the examples show, nothing prevents a whole from having parts that are themselves wholes (with a different UC).

As it was the case for rigidity, in some situations it maybe important to distinguish properties that do not carry a common UC for all its instances, from properties all of whose instances are not intrinsic wholes.

Definition: A property has anti-unity (~U) if every instance of the property is not an intrinsic whole. Of course, ~U implies -U.

We can summarise the notation as follows:

Carrying an UC	ϕ^{+U}	ϕ carries an unity condition
Not carrying an UC	ϕ^{-U}	ϕ does not carry an unity condition
Carrying an anti-UC	$\phi^{\sim U}$	ϕ carries an anti unity condition

Finally, [Guarino and Welty, 2000] defines the notion of External dependence: a property "φ is externally dependent on a property ψ if, for all its instances x, necessarily some instance y of ψ must exist, which is not a part nor a constituent of x." Authors give the example of PARENT being externally dependent on CHILD (one cannot be a parent without having a child), but PERSON is not externally dependent on heart nor on body (because any person has a heart as a part and is constituted of a body).

The extensive work of Guarino contributes to clean-up the theoretical foundations of ontology engineering. Where things become even more useful, is that based on these definitions [Guarino and Welty, 2000] identifies constraints to be verified by the taxonomy so that an ontologist relying on these definitions can check some validity aspects of his subsumption links:

Rigidity constraints: $\phi^{\sim R}$ cannot subsume ψ^{+R}

Identity constraints: ϕ^{+I} cannot subsume ψ^{-I} and properties with incompatible ICs are disjoint.

Unity constraints: ϕ^{+U} cannot subsume ψ^{-U} and $\phi^{\sim U}$ cannot subsume ψ^{+U}

Dependence constraints: ϕ^{+D} cannot subsume ψ^{-D}

The only problem so far is that no tool is available to help an ontologist do that work independently from a formalism, and it can become titanic to apply this theory to large ontologies. IODE from OntologyWorks [Guarino and Welty, 2002] is an interesting integrated tool that is currently being used in several projects.

2.2.5.4 Formal concept analysis for lattice generation

To assists the structuring, semi-automatic methods are of great interest. Formal Concept Analysis is a technique to determine and visualise the correlations of concept and attributes. The models of a domain is composed of formal concepts and their attributes. The formal context is a matrix in which the rows are concepts and columns are attributes; each cell specifies whether a given concept has a given attribute or not. The relationships among formal concepts can then be represented as a lattice where each node corresponds to the maximum set of concepts that possesses a given attribute. An example is found in [Sowa, 2000b] on Formal Concept Analysis for minimal lattices generation by Michael Erdmann, Bernhard Ganter and Rudolf Wille. Boolean attributes are used as differentia, in an example about beverages (alcoholic / non-alcoholic, etc.):

	Attributes				
Concept Types	**non alcoholic**	**hot**	**alcoholic**	**caffeinic**	**sparkling**
Herb Tea	x	x			
Coffee	x	x		x	
Mineral Water	x				x
Wine			x		
Beer			x		x
Cola	x			x	x
Champagne			x		x

Table 3 Boolean attributes of beverage concepts [Sowa, 2000b]

Starting from this table, the algorithm builds the following lattice:

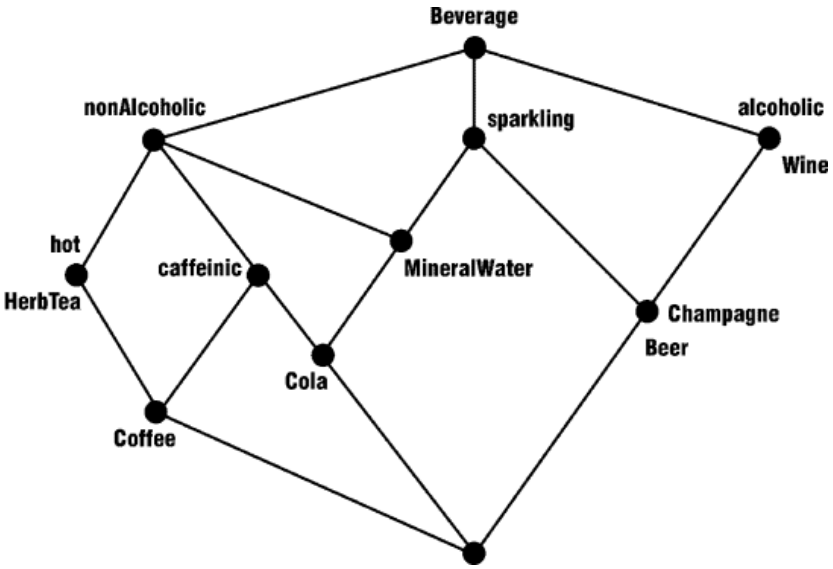


Figure 14 First lattice of beverage concepts [Sowa, 2000b]

The algorithm being incremental and iterative, the addition of two new attributes to distinguish Champagne from Beer can be made afterwards: Made From Grape and Made from grain. Introducing these differences for Beer and Champagne, the algorithm generates a new lattice.

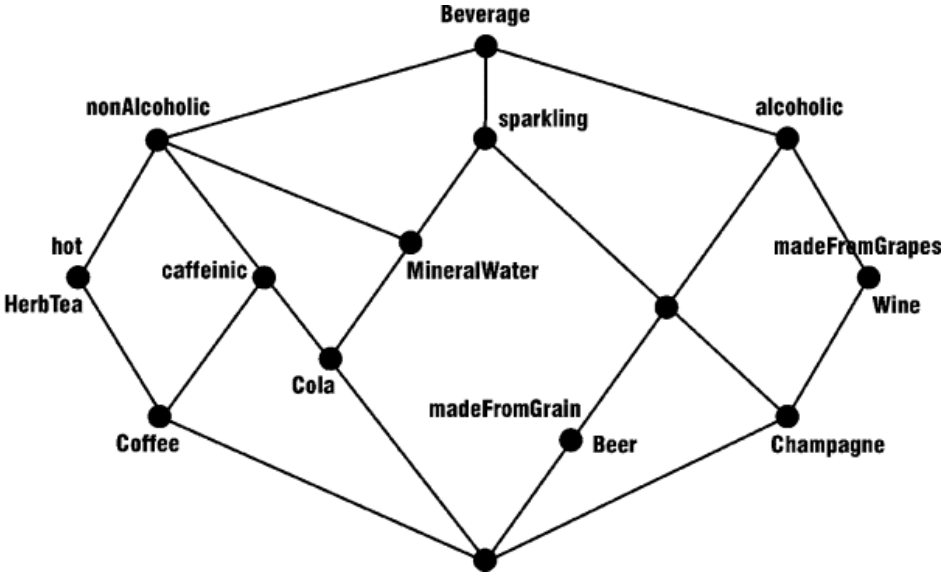


Figure 15 Revised lattice of beverage concepts [Sowa, 2000b]

Yet it still takes a human brain (more precisely a drinker ☺) to see that the Coffee should be declared as made from grain too.

2.2.5.5 Relations have intensions too

In addition to terms corresponding to concepts, the ontologist also gathers terms corresponding to relations. [Bachimont, 2000] declares that the definitions of the relations must be based on the concepts they link. Considering binary relations the author defines them the following way:

- The *semantic signature of the relation*: the concepts linked up by the relation constitute its signature which is part of its definition.
- The *intrinsic semantics of the relation*: it is its *intension*, specified by comparison to other relations according to the differential paradigm. The signature is not sufficient: for example, the relations 'voluntary agent' and 'involuntary agent' besides to have in common their signature, also have in common the parent relation 'agent' while their intensions are different.

In other words, each semantic signature is potentially the root of a differential tree of relations having the same signature and specified according to the differential principle. The semantic signatures also constitute a tree: therefore, we have a relation tree in addition to the concept tree.

We can give the two following examples:

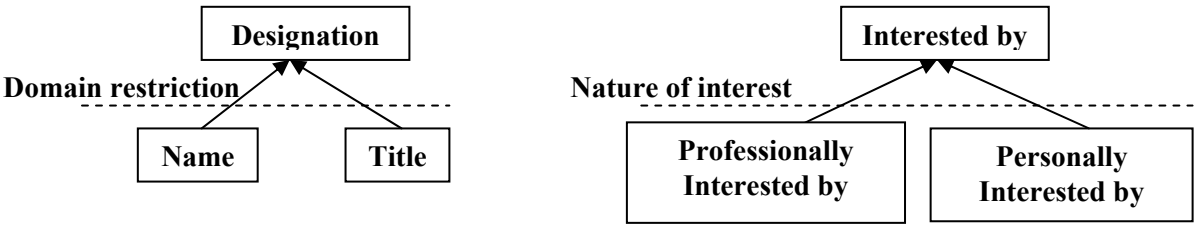


Figure 16 Differentiation in relations

First case the signature is specialised, second case the signature is untouched, but the intension is specialised.

Authors of [Maedche and Staab, 2000] note that "non-taxonomic relations between concepts appear as a major building block in common ontology definitions." They describe an approach for discovering non-taxonomic conceptual relations from text and for facilitating this part of ontology engineering. Building on the taxonomic part of the ontology, their approach analyses domain-specific texts and identifies linguistically related pairs of words. An algorithm for discovering generalised association rules analyses statistical information about the linguistic output. Thereby, it

uses the background knowledge from the taxonomy in order to propose relations. Even if their approach is too weak for fully automatic discovery of non-taxonomic conceptual relations, it does help the ontologist.

Finally, [Staab and Maedche, 2000] claims that the semantics of ontology definitions is mostly void without the specification of axioms, and that axioms are objects, too. Their core idea is to use a categorisation and an object representation of axioms that organise axioms and that provide a compact, intuitively accessible representation. They "reach these objectives through a methodology that classifies axioms into axiom types according to their semantic meaning. Each type receives an object representation that abstracts from particular syntax (as far as possible) and keeps only references to concepts and relations necessary to distinguish one particular axiom of one type from another one of the same type." In their article the authors propose and explore the following categorisation of axioms:

1. Axioms for a relational algebra
   (a) Reflexivity of relations
   (b) Irreflexivity of relations
   (c) Symmetry of relations
   (d) Asymmetry of relations
   (e) Anti-symmetry of relations
   (f) Transitivity of relations
   (g) Inverse relations
2. Composition of relations (e.g. GreatFather = Father o Father)
3. (Exhaustive) Partitions (e.g. Mammal and Fish share no instances)
4. Axioms for sub-relation relationships
5. Axioms for part-whole reasoning
6. Non monotonicity
7. Axioms for temporal and modal contexts

2.2.5.6 Semantic commitment

[Bachimont, 2000] concludes and summarises his opinion on semantic commitment and modelling primitives as follows:

> "The differential principles associated to a node of the ontological tree make explicit in linguistic domain terms what must be understood from the wording of the node. (...) Therefore, it is by respecting these principles that we can ensure that the wording is not merely a linguistic unit with a meaning varying with its use context, but a primitive with an invariable meaning. (...) the differential principles refine and adjust the meaning that users spontaneously attribute to the nodes of the ontological tree (...) We obtain a network in which the position of a node determines its meaning. The meaning defined by the position in the tree is independent from the context. The wording can then be used as a primitive. By respecting differential principles, by committing to their semantics, the nodes of the ontological tree correspond to concepts that can be used as modelling and formalising primitives. We have just defined the semantic commitment on which is based the ontology: set of interpretative prescriptions that must be respected for a wording to operate as a primitive.
>
> Semantic normalising builds one meaning by favouring a special context, the one of the considered task. The semantic commitment brings out an ontology only locally valid, regionally valid in the framework of a domain and a task." [Bachimont, 2000]

2.2.6 Formalisation and operationalisation of an ontology

Continuing on the continuum between informal and formal representation of an ontology, the next stage is to choose a language to capture the content of the intermediary representation we built. The taxonomy is one of them, but the needed expressiveness may be higher.

> "The degree of formality by which the vocabulary of an ontology is specified varies from informal definitions expressed in natural language to definitions stated in a formal language such as first-order logic with a rigorously defined syntax and semantics. Similarly, (...) the uses of ontologies ranged from informal requirements such as a glossary for shared understanding among users to more formal requirements such as interoperability among software tools.
>
> The formality required from the language for the ontology is to a large extent dependent on the degree of automation in the various tasks which the ontology is supporting. If an ontology is a framework for communication among people, then the representation of the ontology can be informal, as long as it is precise and captures everyone's intuitions. However, if the ontology is to be used by software tools or intelligent agents, then the semantics of the ontology must be made much more precise." [Uschold and Gruninger, 1996]

Thus, the degree of formalisation depends on the operationalisation needs. [Uschold and Gruninger, 1996] even proposed four points along what might be thought of as a continuum:
- *highly informal*: the ontology is expressed loosely in natural language;
- *semi-informal*: the ontology is expressed in a restricted and structured form of natural language, greatly increasing clarity by reducing ambiguity;
- *semi-formal*: the ontology is expressed in an artificial formally defined language;
- *rigorously formal*: the ontology is meticulously defined with formal semantics, theorems and proofs.

The final formal degree of the ontology depends on the use intended and it starts from highly informal, as we shall see when integrating already existing formal ontologies. It is important to recognise that the formalisation task does not consist of replacing an informal version by a formal one, but to augment an informal version with the relevant formal aspect needed by the operational system. The purpose is not to take an informal ontology and translate it in a rigorously formal ontology, the purpose is to develop the formal counterpart of interesting and relevant semantic aspect of the informal ontology in order to obtain a documented (informal description possibly augmented by navigation capabilities from the formal description) operational ontology (formal description of the relevant semantic attributes needed for the envisioned system). In the context of a given application, the ontologist will stop his progression on the continuum between informal and formal ontology as soon as he has reached the formal level necessary and sufficient for his system.

The informal version of the ontology is not merely an intermediary step that will disappear after formalisation, the formal form of ontology must include the natural language definitions, comments, remarks, that will be exploited by humans trying to appropriate the ontology. The informal natural language version is important because as stressed by [Mizoguchi and Ikeda, 1997], ontologies have to be intelligible both to computers and humans. This plays an important role for documenting the ontology and an it eases ontology reuse, reengineering and reverse-engineering. Formal languages and logics may be vital for computational implementation, yet the natural language and the terms remain the natural means of access for humans.

This point is also corroborated in [Uschold and Gruninger, 1996] talking about the wording used:

> "Although the text version of the ontology served as the specification for producing code, there was a requirement that it be accessible to non-technical readers. To achieve an appropriate balance between technical precision and clarity, (1) we kept the text definitions relatively informal, (2) equivalent, but more technically precise definitions cast using the primitives in the meta-ontology are used in documentation directly accompanying the code. An ontology should effectively communicate the intended distinctions to humans who design agents. This means that ambiguity should be minimised, distinctions should be motivated, and examples should be given to help the reader understand definitions that lack necessary and sufficient conditions. When a definition can be specified in formal axioms, it should be. In all cases, definitions should be documented with natural language and examples to help clarify the intent."

An important point too, is that one must make the difference between a formalisation needed for ontology engineering purposes (e.g. in the case of Guarino, the additional formalisation is used for coherence checking) and a formalisation needed for operationalisation *i.e.,* for which the formal aspect will be exploited at run-time. This difference is, in my humble opinion, overlooked.

Finally, from the operationalisation point of view, [Bachimont, 2000] explains that formalising knowledge is not sufficient, we must use it in an operational system. A system can only exploit a concept according to the operations or rules that it can associate with it. Therefore, the semantic allowing a system to use a concept is the computer specification of the operations applicable to a concept. By adding a computational semantics to the concepts of the ontology, Bachimont defines the computational commitment. Inferences and computational semantics are currently mainly buried in the code of software. Their intension and intention are not captured yet they play a vital role in the choices of conceptualisation.

To make an ontology computable, it needs to be implemented in a formal language. If this formal language is standardised and if there exist platforms complying to the standard then atleast a minimum set of operation is certified and the computational commitment exists.

[Uschold and Gruninger, 1996] explains that coding the conceptualisation captured in the previous stage in some formal language involves: Committing to the basic set of primitives that will be used to specify the ontology (class, properties, etc.) this set is often called the 'meta-ontology'; Choosing a representation language which support the meta-ontology; Writing the code.

The authors also underline the necessity of minimising the encoding bias: "The important advice here is to not commit to any particular meta-ontology. Doing so may constrain thinking and potentially lead to inadequate or incomplete definitions." Even if the considerations of the formalisation should not pollute the conceptualisation step, it looks like people conceptualising are biased by their background and may be thinking in terms of classes and inheritance much too early in the conceptualisation. It is a very hard task to make abstraction of any previously used language to start a fresh new conceptualisation. [Uschold and Gruninger, 1996] devised their own meta-ontology, using the natural language definitions. "The main terms defined in the Enterprise Meta-Ontology were Entity, Relationship, Role, State of Affairs and Actor. These served as the basis for the formal coding stage."

There exist several families of formalisation languages. A formalism framework provides primitives with a fixed semantic and manipulation operators with a known behaviour. A symbolic system alone means nothing. A formalism provides a symbolic system (syntax, axioms, inference rules, operators...) and the semantic attached to it (rules of interpretation attaching meaning to symbolic expressions). Thus for ontologies, the stake is to find a formalism providing the adequate modelling primitives to capture the aspects of the ontology for which, according to the scenarios, it was deemed relevant to implement a formalisation.

2.2.6.1 Logic of propositions

Logic is the foundation of formalisation languages. It comes from a branch of philosophy (Logic) that analyses inference and tries to provide a formal scientific method of examining or thinking about ideas. Logic develops logical systems (logical languages with authorised manipulations) that are symbolic systems the interpretation of which provides a simulation of human inference. The propositional logic is the base for other logic. From ontology formalisation point of view, it is too limited: propositions are indivisible symbols. The propositional logic only considers relations between propositions without considering the structure and the nature of the proposition. One of the consequences is that it cannot represent the difference between individuals and categories and the relations between individuals. These differences are at the heart of ontologies and thus we need a more expressive language.

2.2.6.2 Predicate or first order logic

In his Organon, Aristotle studied the categorical propositions:

Quantificator + Subject + Copula + Predicate

These are declarative statements (not interrogative or imperative) where the quantificator is "Universal" or "Particular" and the Copula is "Affirmative" or "Negative". Thus Aristotle distinguished four types of propositions:

- **A**: Universal affirmative: Every A is B e.g. Every Man is Mortal
- **E**: Universal negative: No A is B e.g. No Man is Immortal
- **I**: Particular affirmative: Some A is B e.g. Some Man is Blond
- **O**: Particular negative: Some A is not B e.g. Some Man is not Blond

A, E, I, O are the 4 basic "praedicatum" of Aristotle and they are the foundations of an extension of propositional logic called predicate logic or first order logic. The Organon of Aristotle goes further, studying the inferences that can be done with these predicates, and especially the syllogisms: inferences starting from two propositions (premises) to infer one new proposition (conclusion).

There exist 64 types of syllogisms among which 15 are interesting. Among these 15 syllogisms, 4 are at the heart of ontologies as noted by [Sowa, 2000b]:

Barbara	**Celarent**
A: Every animal is material. A: Every human is an animal. ⇒ A: Every human is material.	E: No spirit is a body. A: Every human is a body. ⇒ E: No spirit is a human.
Darii	**Ferio**
A: Every beast is irrational. I: Some animal is a beast. ⇒ I: Some animal is irrational.	E: No plant is rational. I: Some body is a plant. ⇒ O: Some body is not rational.
Principle of inheritance	**Principle of Coherence**

Figure 17 Main syllogisms used for ontologies

The logic of predicate or first order logic (FOL) includes the logic of propositions. The addition of the universal quantificator and of the predicates gives us the ability to differentiate among individuals and categories and to express relations between individuals. For instance, in this logic we can now write: $(\forall x)$ (cat(x) $\supset$ animal(x)) *i.e.,* for every *x* if *x* is a *cat* then *x* is an *animal*, in other words, the concept *animal* subsumes *cat* (the semantic of subsumption is the one of set inclusion).

The language CycL (Cycorp http://www.cyc.com/cyc-2-1/ref/cycl-syntax.html) used for the Cyc ontology is based on the logic of predicates. KIF [Genesereth and Fikes, 1992] is also based on first order logic and was developed as a Knowledge Interchange Format to solve the problem of heterogeneity of languages by proposing a lingua franca for knowledge exchange between independent systems. It is used by the Ontolingua server [Farquhar *et al.*, 1996]

The first order logic is much more expressive than the logic of propositions. However it also illustrates the paradox of the expressiveness [Kayser, 1997]:
- Even if it is more expressive than the logic of predicates, there are things we cannot express, for instance we cannot give properties on relations e.g.: we cannot say "(transitive(R) ≡ ((∀x)(∀y)(∀z) (R (x,y) ∧ R (y,z) ⊃ R (x,z))"
- On the other hand this logic in only semi-decidable *i.e.,* there does not exist one algorithm to determine, in a finite time, if one expression is provable or not.

The knowledge representation languages presented in the next section usually make some restrictions of the expressiveness to keep the expressiveness they desperately need and cut the rest so that the system remains usable.

I shall not go into the details of every formalism developed by the knowledge engineering community. I only give an overview of the formalisms available. These different languages have to be evaluated and compared bearing in mind the trade-off between expressiveness and efficiency, usability and reusability.

2.2.6.3 Logic Programming language

Traditional logic programming languages can be used to formalise knowledge models and knowledge bases. However unlike the following knowledge engineering dedicated languages, they are far less structured to be used for that application. In particular, there is no native distinction between ontological and assertional knowledge. Any piece of knowledge is represented by a set of Horn clauses *i.e.,* a statement of the form *head* ← *body*. A clause with no head is a goal; a clause with no body and no variable in its head is a fact. Other clauses are rules; they are used by inference engine together with facts to derive if goals can be successfully achieved or not.

2.2.6.4 Conceptual graphs

Conceptual graphs (CG) [Sowa, 1984] [Sowa, 2002] come from a merging between existential graphs (Peirce) and semantic networks (Quillian). This formalism was motivated by needs for natural language processing and needs for friendly presentation of logic to human. Graphs have several representations:
- *DF* (Display Form): a graphic representation;
- *LF* (Linear Form): a textual linear equivalent of the DF;
- *CGIF* (Conceptual Graph Interchange Format): for transmission between systems.

A CG is a bipartite oriented graph *i.e.,* there are two types of nodes in the graph (concept nodes and relation nodes) and the arcs are oriented and always link a concept node to a relation node (or vice versa). CGs are existential and conjunctive statements. Relations are n-adic *i.e.,* their arity (valence) is an integer *n* giving the number of concepts they can be linked to. Concepts and relations have a type. The types are primitive or defined. A definition is given by a λ-expression *i.e.,* a graph with formal parameters λ_i that give the definitional pattern. Types of relations also have a fixed valence (giving the number of concepts linked by the relation) and a signature (giving the type of concepts linked by the relation). A conceptual graph is a bipartite graph made of concept nodes and relation nodes. The ontological knowledge upon which conceptual graphs are built is represented by the support, made of two subsumption hierarchies structuring concept types and relation types, a set of individual markers for the concepts, and signatures defining domain and range constraints for the relations. The core reasoning operator of Conceptual Graphs is the computation of subsumption relations between graphs, called specialisation/generalisation relations. It is based on the projection, a graph homomorphism such that a graph *G* subsumes a graph *G' iff* there exists a projection from *G* to *G'*. The projection takes into account specialisation of relations and concepts *i.e.,* nodes of the graph *G* must be of the same type or must be subsumers of the nodes in *G'* they are mapped to.

There are several levels of extensions to CGs starting from simple graphs [Chein and Mugnier, 1997]: there exist an extension for nested graphs and contexts, another for rule graphs for inferences, another for actors enabling procedural attachment, etc. Nested graphs introduce contexts that are concept nodes with a CG inside. They allow, for instance, a CG to be the subject of another CG. The contexts can be nested. Rule graphs allow graphs to describe *IF... THEN* rules to infer new CGs from known facts [Salvat and Mugnier, 1996].

Known platforms implementing CGs are: CoGiTo & CoGITaNT, Notio (API Java), CharGer (CG editor), WebKB (CG in information retrieval), CG Mars Lander (Question-Answer system), Prolog+CG - (object-oriented extension of PROLOG, based on CG implemented in JAVA), Project Peirce (A collaborative project for developing a CG workbench), Synergy etc.

2.2.6.5 Topic Maps

Topic Maps [Biezunski, 2001] is a representation proposed by the librarian community to index electronic documents, manage glossaries, thesaurus and catalogues, enable merging of indexes. There exists an XML language XTopic to exchange Topic Maps.

A Topic reifies a subject in the form of multi-headed link pointing to occurrences of this subject in a mass of documents. The "subject" of a topic is the thing that it is about. Topics are instantiated outside the information sources and they collectively construct a topic map.

Roles are subgroups of occurrences of a topic, for instance there can be a role "mention" and a role "definition". Topics are grouped in classes called "topic types". A topic type is a category to which one given topic instance belongs. Topic types are organised in a subsumption hierarchy. A Topic can have three types of names: a base name / designation name; a display name that can even be a graphic; a sorting name.

Topics can be related together through some association expressing given semantic, an association is a relation constrained to only relate topics together. An example given in [Biezunski, 2001] is an "employment" association that can be used to describe the relationship between a person (employee) and a company (employer).

A Characteristic of a topic is a set of names, occurrences and roles for this topic. These names, occurrences and roles are called the scope of the topic and enable us to define points of view or profiles on a topic e.g.: for a given profile of user, the system will use a given subset of names, occurrences and roles.

Facets are filters on Topics Maps, for instance they enable to extract from a multilingual topic map a precise language.

2.2.6.6 Frame and Objet oriented formalisms

Object Oriented Formalisms [Ducourneau *et al.*, 1998] propose to represent, capture organise and manipulate knowledge through the notion of virtual objects representing real objects. In these formalisms there exist two basic entities: object classes and object instances.

Classes are categories of objects. A class defines the characteristics shared by all the objects of this category. Classes are structured in an inheritance hierarchy defined by the link "a-kind-of".

A class can be instantiated *i.e.,* one can create an object belonging to this class. Instances are final objects instantiated from a class. Instances are linked to their class by a link "is-a". An instance has a unique identifier and attributes. Every attribute has a list of facets giving the value and characteristics of the attribute.

The semantic of inheritance is the one of inclusion of sets *i.e.,* the instances of a subclass are also instances of its super class. Subclasses inherit attributes and facets; they can enrich these definitions by adding new attributes, new facets or by refining constraints. Facets can be declarative or

procedural to specify the nature (type, domain, cardinality, value) or the behaviour of an attribute (default value, daemon *i.e.,* procedures to calculate the value, constraints, filters).

A mutation is the operation of trying to change the class of an object. This operation is at the heart of the classification algorithms of Object oriented frameworks that try to automatically classify instances according to their characteristics.

Points of view can be defined to build different hierarchies of classes capturing different conceptualisations while enabling an object to inherit from all the aspects defines for its class in the different views. Graphic modelling languages (OMT, UML) have been proposed that look like graph-oriented languages.

Object oriented data bases also offer interesting schema definition capabilities and additionally provide efficient data storage and retrieval mechanisms based on object query languages.

Examples of Object oriented systems are: FRL: MIT 70, RLL: Lenat 80, SRL: Fox 78-85, KRL: Xerox, Units, KL-one, Shirka, Smeci Yafool & Y3, Troeps, Arome, Frome, etc. Object oriented data bases include O2, Ontos, ITASCA, Gemstone, Objectstore, Versant, Matisse, Objectivity/DB, etc. OKBC [Chaudhri *et al.*, 1997] : Open Knowledge Base Connectivity is a protocol with an object-based representation language. XOL [Karp *et al.*, 1999] is a light version of OKBC with an XML syntax. FLogic [Kifer *et al.*, 1995] integrates frame-based and object -based languages and first order logic.

2.2.6.7 Description Logics

Description logics [Ducourneau *et al.*, 1998] [Kayser, 1997] are drawing upon predicate logic, semantic networks and frame languages. There again there are two levels: the terminological level where concepts and roles are represented and manipulated and the factual level where assertions and manipulations are made about individuals. Assertions constitute the assertional box (A-Box) while the ontological primitives upon which assertions of the A-Box are built is represented by the terminological box (T-Box). A T-Box is made of a set of concepts being either primitive or defined by a term, a set of roles, and a set of individuals. A concept is a generic entity of an application domain representing a set of individuals. An individual is a particular entity, an instance of a concept. A role is a binary relation between individuals. The description of a role can be primitive or defined. A definition uses the constructors of the language to give the roles attached to a concept and the restrictions of the roles (co-domain).

Concepts are organised in a subsumption hierarchy which is computed according to concept definitions (this definition is reduced to a subsumption link in the case of elementary concepts). The fundamental reasoning tasks in DLs are the computation of subsumption relations between concepts to build the taxonomies and the classification *i.e.,* the automatic insertion of a concept in the hierarchy, linking it to its most specific subsumer and the most general concepts it subsumes.

There exist different languages and families of description logics varying in the set of constructors available:

- *AL* language: minimum language $AL = \{T, \bot, \neg A, C \cap D, \forall r.C, \exists r\}$ where *A* is a primitive concept, *C* and *D* are defined concepts and *r* is a role.
- *FL* et *FL-* of Brachman $FL = \{C \cap D, \forall r.C, \exists r, r|C\}$ where r|C can be written (restrict r C) and introduces a constraint on the co-domain of the role r. $FL^- = \{C \cap D, \forall r.C, \exists r\}$
- *PL1* et *PL2*

A language can then be declined in a family of more expressive languages for instance fo *AL*:

- $ALC = AL \cup \{\neg C\}$ negation of defined concepts
- $ALU = AL \cup \{C \cup D\}$ disjunction of defined concepts

$\bot \equiv C \cup \neg C$ and $C \cup D \equiv \neg(\neg C \cap \neg D)$

- $ALE = AL \cup \{\exists\ r.C\}$ or c-some
  typed existential qualification
  $\exists r \equiv \exists\ r.T$ and $\exists\ r.C \equiv \neg(\forall r.\ \neg C)$
- $ALN = AL \cup \{\geq n\ r, \leq n\ r\}$ or atleast and atmost
  cardinality: minimum and maximum number of values.
- $ALR = AL \cup \{r1 \cap r2\}$ or (and r1 r2)
  conjunction of roles.
- Then these extensions can be composed to create even more expressive languages the biggest one being $ALCNR = ALCUENR$ because $C \cup D \equiv \neg(\neg C \cap \neg D)$ $\exists\ r.C \equiv \neg(\forall r.\ \neg C)$. Of course this expressiveness is at the cost of efficiency.

Examples of systems based on Description logics are: LOOM [Mac Gregor, 1991], Classic, Back, Kris, K-Rep, KL-One, etc. OIL [Fensel *et al.*, 2000] is an extension of the XML language RDF(S) - (see chapter 3), it is based on descriptions logics.

2.2.6.8 Conclusion on formalisms

A comparison of several of several of the knowledge modelling languages is given in [Corcho and Gómez-Pérez, 2000].

Knowledge modelling languages that offer automatic classification (e.g. description logics) may be useful to assist building ontology and allow maintenance updates, but the paradox of formalisation is that to obtain this assistant the systems need formalisation (formal definitions) that may rely on additional notions that will have to be structured too, and so on.

Most of the knowledge engineering languages provide primitives to distinguish the ontological knowledge from the assertional knowledge, and have dedicated primitives for the description of the taxonomic structure of the ontology. The differences come when additional granularity require, for instance, formal definitions. Three possibilities are used by formalism: rules, formal definitions, and dedicated primitives to denote algebraic properties of relations.

The definition of concepts enables reasoning over concepts, for classification and logical equivalence inferences. The definition of concepts is an built-in characteristic of Description Logics, to allow concept classification consisting in the deductive inference of the most specific concepts which subsume a given concept and the most general concepts it subsumes. The definition of concepts is also a feature of the original Conceptual Graphs model defined in [Sowa, 84]. Finally, in Logic Programming, partial definitions of concepts (sufficient conditions) may be represented as rules.

Rules are indeed an alternative and explicitly capture inferences that can be used to factorise knowledge in the ontology and discover implicit knowledge in an assertion. This factorisation generally consist in capturing patterns that are the logical consequence of another pattern. For instance: if a person *x* is a male and the child of a person *y* then *x* is the son of *y*. If a formalism does not provide primitives to express algebraic properties of relations (transitive, symmetric and reflexive) or inverse relations, rules may be used for that purpose. For instance, for symmetry: if a person *x* is married with a person *y* then *y* is married *x*; this sounds like a tautology, but if it is not explicitly said to the system then symmetry is lost and the corresponding implicit knowledge is not found at the assertional level. Logic Programming enables the expression of rules, as Horn clauses. Aside from the concept expressions constructs, most Description Logics are also provided with rules, similar to Horn clauses. Finally, the classic Conceptual Graphs formalism is not provided with rules, but an extension is proposed in [Salvat and Mugnier, 1996] to handle graph rules with graphs as head and body.

2.2.7 Reuse

The reuse of ontologies is both seductive (it should save time, efforts and would favour standardisation) and difficult (commitments and conceptualisations have to be aligned between the

reused ontology and the desired ontology). But [Guarino, 1997] is right saying that "a concept may be 'relevant' for a particular task without being necessarily 'specific' of that task". Therefore, reuse should be possible and pursued. "One should try to integrate as much as possible existing ontologies in one's ontology." [Fernandez *et al.*, 1997]

The approach taken in the project described in [Bernaras *et al.*, 1996] is interesting: "The feasibility of knowledge reuse is investigated by creating ontologies for particular domains and reusing them for different tasks or applications. This can be called domain-oriented reuse." The authors say that modularization and hierarchical organisation are good for usability and reusability. However, theoretical distinctions for structuring ontologies can be practically difficult to operationalise and maintain.

> "Ontologies are developed for the purpose of reusability and shareability of knowledge, and reusability is directly linked with generalisation, *i.e.*, generic concepts are usually more reusable than specific ones. The desired scope of reusability is a very important decision that has to be taken before an ontology is designed. Although it is true that generic concepts are in general more reusable, the reuse of generic concepts for specific applications may involve, in certain cases, a big design effort to translate the generic concepts into specific ones. This effort has to be seriously considered during the design of an ontology, and compared with the effort of reusing, for example, an ontology built of specific concepts belonging to related applications." [Bernaras *et al.*, 1996].

[Fernandez *et al.*, 1997] say "ontologies are built to be reused or shared, anytime, anywhere, and independently of the behaviour and domain of the application that uses them." This is quite contradictory with the idea of building an ontology in the context of a scenario. In fact, there exist large general ontologies that can be reused to create the definitions instead of starting from scratch. These ontologies have not been built in a scenario fashion, but with an encyclopaedic perspective.

[Fernandez *et al.*, 1997] advocate two steps in reuse:
- *Inspect meta-ontologies* to select those that better fit your conceptualisation while guaranteeing that the sets of new and reused definitions are based upon the same set of basic terms.
- Find out which *libraries of ontologies* provide definitions of terms whose semantic and implementation is coherent with the terms identified in your conceptualisation. If the meta-ontology upon which those terms have been built is different of the meta-ontology for your ontology, you should check the existence of translators to transform definitions into your target language.

Translation is a very delicate point, especially because it is limited to the formal part of the ontology and it is only one of the problems raised by the reuse of ontologies (different commitments may be difficult to conciliate). But even the formal translation has limitation, for instance translation into less expressive languages means that translation might be incomplete.

An idea discussed in the community is the possibility of building libraries of ontologies. Then, comes the problem of indexing, characterising and comparing the relevance of ontologies in the library for a given new problem. One proposition is to base the classification of ontologies on the description of the problems, context, perspective that were in the mind of their creators. [Uschold and Gruninger, 1996] proposed that the ontologies be distinguished by their corresponding competency questions (these are queries arising in the scenarios and placing expressiveness requirements on the envisioned ontology). The competency questions would be a surrogate to index ontologies in the library.

[Guarino, 1997] defends the thesis of the independence of domain knowledge:

> "This thesis should not be intended in a rigid sense, since it is clear that – more or less – ontological commitments always reflect particular points of view (for instance, the same physical phenomenon may be described in different ways by an engineer, by a physicist or by a chemist); rather, what I stress is the fact that reusability across multiple tasks or methods can and should be systematically pursued even when modelling knowledge related to a single task or method: the more this reusability is pursued, the closer we get to the intrinsic, task-independent aspects of a given piece of reality (at least, in the common sense perception of a human agent)."
>
> [Guarino, 1997]

On the other hand, [Bachimont, 2000] observed that it emerges from practice that it is always possible to adapt an ontology, but never possible to reuse it as it is. [Uschold and Gruninger, 1996] also remarked that during both of the capture and coding processes, there is the question of how and whether to use all or part of ontologies that already exist. "It is easy enough to identify synonyms, and to extend an ontology where no concepts readily exist. However, when there are obviously similar concepts defined in existing ontologies, it is rarely clear how and whether such concepts can be adapted and reused." [Uschold and Gruninger, 1996]

It is important to notice that here again, the natural language version of the ontology and documentation of the ontology are of use. They enable humans to understand the ontology and can be integrated in the initially analysed corpus to enrich the terminological study, and suggest other aspects that were not captured in the formal version, but still implicit in the natural language definitions and interesting in a new application context.

Following [Sowa 2000b] we can say that each of the three basic fields at the heart of ontology-oriented modelling Logic, Ontology, and Computation, presents a different class of problems for knowledge sharing and thus for ontology sharing and mapping between different ontologies:

- *Logic*: "Different implementations support different subsets and variations of logic. Sharing information between them can usually be done automatically if the information can be expressed in the common subset. Other kinds of transfers may be possible, but some of the information may be lost or modified." [Sowa 2000b]. Thus the very first problem is the differences of expressiveness between the formalisms. Loss will occur as soon as a translation lacks the needed expressiveness in the target language.
- *Ontology*: "Different systems may use different names for the same kinds of entities; even worse, they may use the same names for different kinds. Sometimes, two entities with different definitions are intended to be the same, but the task of proving that they are indeed the same may be difficult or impossible" [Sowa 2000b]. We saw that the granularity and the scope of the ontologies were very much dependent on the application scenario. Thus an ontology may be interesting for its domain, but hardly reusable because it was tuned for a completely different application. However, the natural language definitions, comments and documents of the ontology can still be exploited since they represent a highly relevant corpus for text analysis.
- *Computation*: "Even when the names and definitions are identical, computational or implementation side effects may cause the same knowledge to behave differently in different systems. In some implementations, the order of entering rules and data may have an effect on the possible inferences and the results of computations. Sometimes, the side effects may cause a simple inference on one system to get hung up in an endless loop on another system" [Sowa 2000b]. The importance of computational commitment is vital since it is not respected it will spoil the logical and ontological consensus. The inference intensions should be captured, documented and publish to enable exchanges, consensus and checks. Rule languages are, in some way, a means to do it. Standard APIs and standards documentation including test bases are currently the best way to ensure systems are behaving the same way.

We have seen the importance of keeping the natural language definitions and comments in the formalised version; this is a first and extremely important step in documenting the ontology and its life. "An ontology should effectively communicate the intended distinctions to humans who design agents. This means that ambiguity should be minimised, distinctions should be motivated, and examples should be given to help the reader understand definitions that lack necessary and sufficient conditions. (...). In all cases, definitions should be documented with natural language and examples to help clarify the intent." [Uschold and Gruninger, 1996]

The design rationale of the ontology should be captured because it explains what motivated its actual state and helps people understand and maybe commit or adapt it. All important assumptions and design choices should be documented An important use of the documentation is also the maintenance or reuse processes. "The absence of a sound documentation is also an important obstacle when you reuse/share ontologies already built." [Fernandez *et al.*, 1997] their methodology (METHONTOLOGY) includes the documentation as an activity to be done during the whole ontology development process.

Documenting is not only interesting for designers, the document can prove to be a strong asset to encourage the appropriation of the ontology by the users of the system exploiting this ontology and reuse by the designers of other systems.

3 Structured and semantic Web

To communicate through silence is
a link between the thoughts of man.
— Marcel Marceau

The World Wide Web is now a huge repository of information resources which silently grows link after link. As we saw in the section 1.5 on information retrieval, search mechanisms lack the understanding needed to evaluate the relevance of a resource against the needs expressed in a query. In the current Web, information resources are machine readable, but not machine understandable. The envisioned Web where the semantics of the resources would be accessible to software for reasoning is called the Semantic Web. The need of semantics is obviously directly linked to the ontological considerations of the previous chapter.

Just like previous web and internet technologies entered organisation internal infrastructure, the semantic web technologies are likely to come inside. In fact, in the past, new technologies were usually developed at the scale of some organisations and then expanded to word-wide coverage. Then they were finally re-introduced as local solutions and at a smaller scale inside organisations because the large scale diffusion had made them affordable standards largely tested and proven. In the first part of this chapter, I shall survey such small-scaled pioneer projects that prefigured the semantic web by considering ontology-based annotation for resources retrieval. The brief second part will extend the set of definitions used in this Ph.D. with some web-related notions. The third part will detail the foundations of the Semantic Web, as they were imagined by the W3C. The fourth part will summarise this view and present current works and perspectives. Finally, I shall discuss some aspects of the current trends.

3.1 Beginnings of the semantic web

We have seen information retrieval issues and approaches and then we have seen ontology issues and approaches; it is clear that there is an interest in considering a field merging them; this field started long before the semantic web concept appeared. In the first part, I shall briefly survey pioneer projects that prepared the ground for the semantic web. In the second part, I shall summarise the possible characteristics of an ontology-based information retrieval system.

3.1.1 Inspiring projects

3.1.1.1 SHOE

Historically SHOE [Heflin *et al*., 1998] [Luke and Heflin, 2000] was one of the first languages to merge ontologies and Web markup languages. It stands for Simple HTML Ontology Extension. It was developed as an extension of HTML in order to incorporate semantic knowledge in web pages; later it was adapted to XML. In SHOE, an ontology consists of a taxonomy of concepts, simple ground relationships and inference rules. Multiple inheritance between concepts is allowed and relations are n-ary predicates defined with their signatures. SHOE designers purposefully made it simple, arguing that the cost of comprehensive semantic expressiveness becomes overwhelming in very large domains such as the Web. Each concept or relation has a prefix that identifies its source ontology and thus a new ontology can extend existing ontologies through specialisation. The ontology provides the conceptual vocabulary to embed semantic annotations in the web pages and formulate queries. The taxonomy is used to generalise or specialise a query. Inference rules allow logical reasoning in a similar way to Horn clauses without negation in the head nor in the tail of the rule, and with variables universally quantified. SHOE introduces several built-in data types like number, date, string, etc. As the application domain is the open Web, assertions are taken as claims and not as truth statement; they can concern the page they are embedded in or other resources.

3.1.1.2 Ontobroker

Ontobroker or On-2-broker [Fensel *et al.*, 1998] [Fensel *et al.*, 1999] was also one of the first projects intended to improve retrieval of information on the World Wide Web. It uses Flogic (Frame Logic) as the knowledge representation formalism to represent the ontology and to express the annotations of documents. FLogic integrates frames and first order predicate calculus. It accounts in a declarative fashion for most of the structural aspects of object-oriented and frame-based languages, such as object identity, complex objects, restriction of the value of a property, inheritance, polymorphic types, query methods, encapsulation, and others; properties are declared inside classes. It is possible to define inference rules using logic connectors (AND, OR, NOT). The rule body is the conjunction of elementary expressions. Variables can be existentially or universally quantified in the body and are universally quantified in the head of the rule. It is possible to add semantic information directly inside HTML tags to avoid duplicate information in data and metadata. Wrappers were also developed for automatic extraction from stable information sources. A crawler searches for annotations inside Web pages and gathers them in a centralised repository. The ontology is also centralised and cannot be extended by a user. Query formalism is Flogic-oriented, but tools are provided to allow user-friendly search and ontology browsing.

3.1.1.3 Ontoseek

OntoSeek [Guarino *et al.*, 1999] is dedicated to product catalogs and yellow pages on the web. It uses structured content representations coupled with linguistic ontologies to increase both recall and precision of content-based retrieval. Queries and resource descriptions are represented in Lexical Conceptual Graphs, simplified variants of Conceptual Graphs [Sowa, 1984] where labels of concepts and relationships are lexical items *i.e.,* the terms such as "has-function" or "part-of" are forbidden. OntoSeek uses the Sensus ontology [Knight and Luke, 1994] that is composed of 50 000 concepts that results from merging WordNet's thesaurus into the Penman top-level ontology. Such a large ontology allows users to use arbitrary natural language for resource description and query elicitation. Content matching then reduces to conceptual graph matching, where individual nodes and arcs match if the ontology indicates that a subsumption relationship holds between them. Authors point out the need for further inference mechanism: inverse relations such as *child* and *parent*; explicit mutual disjointness for two children of a concept; a term definition mechanism in the ontology, provided with description-based subsumption and automatic classification.

3.1.1.4 ELEN

ELEN [Chevallet, 1992; 1994] is one of the ancestors of knowledge-based information retrieval systems. It used Conceptual Graphs to index and retrieve software source code in order to assist code recycling. The indexing of each piece of code was carried out in a manual way to provide one or more conceptual graphs describing its functionality. Matching is based on the notion of partial projection associated with a probabilistic measure in]0,1[where $P(D \rightarrow Q) = 1$ if there exists a projection Π of Q on D. ELEN proposes a mechanism of search optimisation based on the construction of the signature of graphs summarising their content.

3.1.1.5 RELIEF

RELIEF [Ounis and Pasca, 1998] is also a pioneer, as it focuses on retrieving documents annotated by conceptual graphs and according to different views (structural view, a symbolic view and a spatial view). Annotations are pre-processed and much of the classical projection work is replaced by the construction of a complex inverted index file that provides for a given arc all the documents which are described by a conceptual graph containing such an arc. In addition, all the specialisations are pre-computed and stored. As a result, the matching of a query is reduced to filtering the candidate arcs corresponding to partial projections, allowing for a polynomial matching function. RELIEF handles algebraic properties of relations like symmetry, transitivity, and inverse introducing new entries in the inverted file.

3.1.1.6 LogicWeb

LogicWeb [Loke and Davison, 1998] divides the web into two layers: classic web page content at the first level, and the links between these pages at a logical layer for more flexibility. Thus LogicWeb proposes the use of logic programming to improve both Web site structuring and browsing and searching. Web pages are viewed as logic programs, consisting of facts and rules. The use of logic programming makes it possible to define relationships between web pages and express complex information about pages. The content of the logical layer is created by default heuristics extracting facts from the web page (e.g. Links) and code included in special tags <LW-code> … </LW-code>. The logical layer allows users to build logical structures (semantic networks, hierarchies, etc.) inside a page, to type links, to call procedures in order to compute destination pages and to use meta-programming. The system requires a dedicated Web client to implement the logical layer functions.

3.1.1.7 Untangle

Untangle [Welty and Ide, 1999] [Welty and Jenkins, 2000] is a knowledge-based card catalogue system. It belongs to the domain of digital libraries that extend the traditional librarian techniques to manipulate digital documents over a network. Untangle is a knowledge-based information system which uses semantically enriched information spaces to improve browsing and querying. It combines SGML for specifying tag syntax and description logic to enhance and extend retrieval capabilities. It relies on an ontology where persons, organisations, conferences, etc. are first-class citizens with attributes and relations of their own. It is a change in such card catalogue and bibliographic systems where ontologies are usually limited to books and shelves, with people or organisations as simple free attributes. The Untangle query and retrieval mechanism is based on the representation of a document as an individual instance of a hierarchy of concepts using the CLASSIC description logic [Brachman *et al.*, 1989]. CLASSIC enables the definition of structured concepts in the form of frames, the description of their taxonomy, the creation and manipulation of individual instances of such concepts, and inference such as inheritance, algebraic properties of relations, automatic classification based on formal definition, etc. The ontology is strongly formal (the classes are defined, constraints and inference rules are available) to allow reasoning over the notions defined in the ontology. Inference capabilities enable automatic augmentation of the data mined from existing bibliographic databases through reasoning.

3.1.1.8 WebKB

WebKB [Martin and Eklund, 1999; 2000] is a web-based set of tools allowing its users to represent knowledge to annotate web resources and propose retrieval mechanisms using conceptual graphs. The inference engine exploits subsumption relations to compute specialisation relations between a query graph and an annotation graph. WebKB annotations can be inserted anywhere in a web document by using <KR> ... </KR> tags and the language used is an hybrid of formalised English and conceptual graph linear form. WebKB enables the use of undeclared types and automatically inserts them in the ontology according to the way they are used (signatures of the relation types in which they are used). WebKB is used to improve search, browsing and automatic document generation. The system relies on a top-level ontology of basic concepts and relation types coupled with the natural language ontology WordNet merged at the top-level. The WebKB ontology is formalised as a conceptual graph support made of a concept type hierarchy and a relation type hierarchy. WebKB succeeds to the earlier system CGKAT [Martin, 1996].

3.1.1.9 CONCERTO

CONCERTO [Bertino *et al.*, 1999][Zarri *et al.*, 1999][Armani *et al.*, 2000] stands for CONCEptual indexing, querying and ReTrieval Of digital documents; it is dedicated to the annotation and retrieval of textual documents in the domains of biology and publishing. The annotations are represented in the Narrative Knowledge Representation Language (NKRL) and XML is used as a data interchange format. NKRL is close to frame-based languages and is based on definitional components (concepts), enumerative components (instances), and relations upon which templates (defined concepts) are constructed. The ontology of CONCERTO consists of both a hierarchy of concepts and a hierarchy of templates. Documents are annotated by analysing the output of Natural Language Processing methods. Queries are search patterns *i.e.,* NKRL patterns including variables and constraints. The inference search engine analyses the query to find other possible forms. Then, the search engine matches the different patterns against the templates annotating documents.

3.1.1.10 OSIRIX

OSIRIX [Rabarijaona *et al.*, 1999; 2000] stands for Ontology-guided Search for Information Retrieval In XML-documents. It is a tool proposing ontology-guided search in XML documents applied to corporate memory consultation. Taking into account the advantages of the World Wide Web and of ontologies for knowledge management, OSIRIX relies on XML meta-language for corporate knowledge management. The knowledge models that guide the search are CommonKADS expertise models, represented in standard CommonKADS [Breuker and Van de Velde, 1994] Conceptual Modelling Language (CML). This provides an ontology, from which OSIRIX generates adequate XML elements constituting a DTD. A CML concept and its properties are mapped into XML elements. The system can translate concepts, *n*-ary relations, attributes, cardinality constraints, CML relations and expressions. Based on this DTD, the authors create their XML documents, by respecting the specifications; the fact that the XML elements are linked to knowledge models makes them ontological annotations in the documents. Using the DTD, the validation of a document allows the company to make sure that this document can constitute later on a valid answer and for OSIRIX such a validation facilitates the work of data extraction. In order to answer the user's requests, the system seeks in the ontological part of the documents, if an ontological answer is present or not. The ontological filtering engine finds the documents answers among the candidate documents. If the system does not find exact answers, it can exploit the ontology to seek approximate answers: it can exploit the concept hierarchy to find an answer corresponding to sub-concepts of the concept appearing in the initial request.

3.1.2 Summary on ontology-based information systems

From the above examples, I would first underline a transversal need for a formalism providing both structure definition primitives and semantic definition primitives. Structural symbols and rules provide a symbolic system on which semantic primitives and rules are based; structural and semantic aspects of the framework are a ground consensus above which knowledge models can be made explicit and exploited. The structural and semantic primitives can be provided in a complete formalism (e.g.: conceptual graphs, description logics, object-based languages, etc.) or provided by two complementary components (e.g. XML coupled with a knowledge modelling methodology like CommonKADS [Breuker and Van de Velde, 1994]).

Then, there are four recurrent functionalities:

- *underlying knowledge models elicitation and maintenance*: All the systems require facilities to capture and maintain models of the domains related to their application field. The taxonomic structure of the intensions of the ontological primitives used for metadata elicitation is an absolute must. Additionally, formal definitions and constraints allow classification inferences that assist ontology building. The same inferences may be use during maintenance updates triggered by the extensions suggested by users during their day-to-day interactions.
- *metadata / annotation generation and indexing*: Annotation generation can be manual when it is part of the activity of the users (e.g. librarian). It can also be partially derived using clues from the use context and metadata generating heuristics usually based on pragmatic knowledge. Finally, it can result from complex natural language processing tools exploiting natural language ontologies. Obtained annotations provide an index from which an inverted file can be produced, anticipating and accelerating forthcoming use.
- *query-based search capabilities*: The annotation and query matching usually relies upon the assumption that for a document D annotated by the metadata/annotation A to be relevant to a query Q, $A \rightarrow Q$ must hold. The operationalisation of $A \rightarrow Q$ depends on the language chosen for the representation of queries and of annotations. In Conceptual Graphs, matching is operationalised by the projection operator looking for graphs G_A representing annotations such that $G_A \leq G_Q$ holds where G_Q is a graph representing the query. In description logics, systems build a dedicated

operator looking for assertions A_A representing annotations such that $A_A \leq A_Q$ holds where A_Q is an assertion representing the query. In logic programming, an annotation is represented by a fact F_A and a query by a goal G_Q; the matching succeeds if the logical engine manages to prove G_Q using a fact F_A and the rules that formalise the ontological knowledge.

- *browsing and search capabilities through dynamically structured views*: Browsing the ontological knowledge consists of the exploration of the domain notions and of the attached vocabulary. Taxonomies of notions are a natural way to discover a conceptualisation and to find an entrance point to start querying an information system (in a Yahoo-like fashion). The browsing taxonomies can be directly provided from the declaration of elementary concepts or computed from the definition of complex concepts. Additionally, using the annotations, dynamic structured views of the knowledge base can be computed to browse the documents; such views can be obtained by automatic classification of annotations or exploiting link typing for instance.
- *inference for knowledge and query refinement*: Inferences allow implicit knowledge discovery, but require formalisation in the form of formal definitions or inference rules. Formal definitions and rules capture logical equivalence, implications and production rules to augment assertions by making explicit some of the implicit knowledge they include. The elementary inference is the generalisation / specialisation that allows the matching process to augment its recall by taking into account the specialisation of a required type. It is also used for query reformulation and refinement to generalise failed queries, specialise queries generating too much results, or suggest alternative queries based on neighbourhood in the taxonomy. Additional formal definitions allow the systems to propose alternative expression of knowledge used in queries or assertions. Rules produce new facts from the assertions.

3.2 Notions and definitions

Before I go any further in the exploration of the semantic Web, I complete the set of definitions used in this work

internet (technology)	network technology that enables interconnection of networks, the most well-known instance being the *Internet*. An internet network is decentralised by design: each one of the online computers (called hosts) is independent. This technology is based on the TCP/IP (Transmission Control Protocol/Internet Protocol) suite of communication protocols used to connect hosts.
Internet	the global network connecting millions of computers to exchange data and services. It is based on the internet technology and therefore, it uses TCP/IP, making it the de facto standard for transmitting data over network.
intranet	another class of *internet* instances; intranets belong to an organisation, and are accessible only by the organisation's members or authorised users. Even if an intranet is logically only internal to an organisation, physically it can span the globe, as long as access is limited to a defined community. || Also used to denote a corporate web, an internal web - see *intraweb*.
extranet	intranet that is partially accessible to authorised outsiders. An extranet provides various levels of accessibility to outsiders and the users' identity determines which parts of the extranet can be accessed.
web (technology)	an unstructured client/server network that uses HTTP as its transaction protocol networks, the most well-known instance being the *World Wide Web*. It supports access to documents that are formatted in HTML (HyperText Markup Language). HTML allows the formatting of documents, the creation of links to other documents, as well as the insertion of graphics, audio, video files, etc. There are several applications called browsers that make it easy to access a web.
(World Wide) Web	web comprising all HTTP nodes on the *Internet*.
intraweb	web comprising all HTTP nodes on an *intranet* || syn. corporate web, internal web, and also often called *intranet*.
extraweb	web comprising all HTTP nodes on an *extranet*.
metadata	data about data. The distinction between "data" and "metadata" is contextual to considered task and application, and the same data can be interpreted in both ways. On the web, metadata are used to describe Web resources and structure them.
semantic web	extension of the classical web technology to allow the description and sharing of semantic metadata about and together with information resources.
The Semantic Web	semantic web spanning on the whole *Internet*; extension of the current World Wide Web.

Table 4 Definitions used in corporate semantic webs

The internet technology was used and developed to connect distant machines, but it was soon adopted to build local networks too. The intranet enables an organisation to setup internal e-mail, databases front-end applications, IP phone, etc. by connecting the different hosts (servers, personal computers, etc.) to make available information and services.

The advent of information society led organisations to build intranets that are becoming corporate nervous systems. They memorise critical pieces of information and irrigate the organisational entities with them. What I call an intraweb is the application of web technologies, internally, to provide news, forms, document repositories, web front-end to software, FAQs, helpdesks, etc. with the only requirements of having on the user's machine one of widely available web browsers.

Intranets were first mentioned in the beginning of 1995, but it quite often denotes an intraweb: just like people quite often mix up 'Internet' and 'Web', the intranet word is used to denote an intranet with web services. In this document, I shall use the definitions stated in Table 4.

3.3 Toward a structured and semantic Web

The Semantic Web is not a separate Web, but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in co-operation.
— Tim Berners-Lee

The World Wide Web was originally built for human consumption, and although everything on it is machine-readable, this data is not "machine-understandable". Because of that, it is hard to automate things on the Web, and because of the volume of information it is not possible to manage it manually. We saw that the first need to move toward a semantic Web is to provide a supporting formalism providing a consensual way to define structure and semantics. To address these aspects, the W3C respectively recommended XML and RDF(S) that I shall describe in the following sub-sections.

3.3.1 XML: Metadata Approach

The Extensible Markup Language[1] (XML) is a description language recommended by the World Wide Web Consortium for creating and accessing structured data and documents in text format over internet-based networks. XML is a meta-language used to build languages to describe structured document and data; it can be considered as an SGML light for the Web. It comes with a set of tools and API to parse and process XML files; many of these tools are freely available on the net, but at the same time the format is being more and more available in commercial applications too.

XML documents contain their own structure within themselves and parser can access it and retrieve data from it. The main components of this structure are elements (markup), attributes of elements and text (called PCDATA). The syntax of the languages uses start and end tags to mark up information elements (for example <name> and </name> in Figure 18). Elements may be further enriched by attaching name-value pairs called attributes (for example, country="FR" in Figure 18). Its simple syntax is easy to process by machine, and has the attraction of remaining understandable to humans. XML makes it possible to deliver information to software components in a form that allows automatic processing after receipt and therefore, distribute the processing load. It is also likely to become a de facto standard, and therefore, a good candidate to exchange data and build a co-operation between heterogeneous and distributed sources. XML is said extensible because one can define new tags and attribute names to parameterise or semantically qualify data and documents. Structures can be nested to any level of complexity, so database schemas or object-oriented hierarchies can be represented.

The set of elements, attributes, entities and notations that can be used within an XML document instance can optionally be formally defined in a document type definition (DTD) embedded, or referenced, within the document. The DTD gives the names of the elements and attributes, the allowed sequence and nesting of tags, the attribute values and their types and defaults, etc. The main reason to explicitly define the language is that documents can be checked to conform to it. Therefore, once a template has been issued, one can establish a common format and check whether or not the documents are valid. Figure 19 presents a DTD corresponding to the XML example of Figure 18. Unfortunately the semantics of the tags cannot be described in a DTD. However, if a software knows the semantics, it can use the metadata and infer from them to assist its users. The semantics must be

[1] http://www.w3.org/XML/

shared to allow co-operation among applications and unambiguous exchanges. By describing the meaning of the actual content, structure description will help an application find relevant information and enable matchmaking between producers and consumers.

```
<contact_details>
 <name>INRIA-Sophia</name>
 <address country="FR">
  <street>2004 Route des Lucioles</street>
  <city>Sophia Antipolis</city>
  <postal>06902</postal>
 </address>
 <phone>04-92-38-77-00</phone>
 <last_contact>2002-06-11</last_contact>
</contact_details>
```

Figure 18 XML Sample

```
<!DOCTYPE contact_details [
 <!ELEMENT contact_details (name, address,
    phone, last_contact)>
 <!ELEMENT name (#PCDATA)>
 <!ELEMENT address (street, city, postal)>
 <!ELEMENT phone (#PCDATA)>
 <!ELEMENT last_contact (#PCDATA)>
 <!ELEMENT street (#PCDATA)>
 <!ELEMENT city (#PCDATA)>
 <!ELEMENT postal (#PCDATA)>
 <!ATTLIST address country CDATA #REQUIRED >
]>
```

Figure 19 DTD Sample

XML Schema[1], a new recommendation of W3C, is an XML language that can now replace DTDs. It does not enable us to describe semantics either, but it does allow to type documents and data. Figure 20 shows an example equivalent to Figure 19 with the phone number and the last contact date constrained by a type.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="contact_details">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1"/>
     <xs:element name="address" minOccurs="0" maxOccurs="1">
      <xs:complexType>
       <xs:attribute name="country" type="xs:string" use="required"/>
       <xs:sequence>
        <xs:element name="street" type="xs:string" minOccurs="1" maxOccurs="1"/>
        <xs:element name="city" type="xs:string" minOccurs="1" maxOccurs="1"/>
        <xs:element name="postal" type="xs:string" minOccurs="1" maxOccurs="1"/>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
     <xs:element name="phone" type="typePhone" minOccurs="1" maxOccurs="1" />
     <xs:element name="last_contact" type="xs:date" minOccurs="1" maxOccurs="1" />
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:simpleType name="typePhone">
   <xs:restriction base="xs:string">
    <xs:pattern value="\d+(-\d+)+"/>
   </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

Figure 20 XML Schema Sample

[1] http://www.w3.org/XML/Schema

XML Schema introduces XML Data types which are simple data types such as string, date, number etc. The value of elements can be constrained, specifying patterns based on regular expressions. It is also possible to define complex types with nested elements and attributes and to derive new types from existing ones.

In addition, XML uses namespaces to enable modularisation and sharing of documents and structures. A "namespace" is literally the identification of a source of names for tags. Namespaces enable to identify a specific document structure definition with a unique name *i.e.,* URI and to prefix every tag with this unique name. Thus using namespaces it is possible to distinguish tags having the same names, but coming from different DTDs or XML Schemas.

To explore and manipulate the XML structure, the first requirement is to have the ability to define how to select and point on a given part of the XML structure. Therefore, there is an immediate need for languages to describe paths in XML documents as well as hypertext links.

XPath[1] is a path language that enables to describe a path in an XML structure, express a selection pattern and retrieve element values. A path is composed of steps such as `/book/introduction` which denotes the *introduction* child tags of the *book* element at the root of the document. Paths can include conditions for instance `/book/chapter[contains(child::title,'survey')]/body` selects the *body* of the chapters the title of which contain the string "survey"; the result is the set of *body* nodes satisfying this selection pattern. The path and conditions are expressed on axis that are navigation directions from a node to another, e.g.: `preceding`, `following`, `ancestor`, `child`, `attribute`, etc. Predefined functions (`last()`, `position()`, `count()`, `contains(...)`, `concat(...)`, `starts-with(...)`, etc.) are used to build complex selection paths and manipulate values.

XPointer[2] enables to define a fragment identifier for an URI reference that locates a resource using an XPath. It aims at extending the XPath model to express ranges (*i.e.,* portions of documents with start and end points such as the result of a mouse-driven selection in a document) and at enabling location into a document by string matching.

XLink[2] is the formalism describing (hyper-) links between XML documents, such as hypertext links. Links are described by means of attributes that are inserted in documents. The XLink specification relies on a set of predefined attributes in a specific XLink namespace. The language enables external links and *n*-ary links. External links may be defined outside a participating document, without modifying the source. XLink also enables to associate some semantics to links and to parameterise the behaviour of the link and when it is activated.

XML tags describe a structure of data, rather than the presentation. Content structure and display format are completely independent. The eXtensible Stylesheet Language[3] (XSL) is used for expressing stylesheets, which have document manipulation capabilities beyond styling. Thus a document can be viewed differently and transformed into other documents so as to adapt to the need and the profile of the agents and the users while being stored and transferred in a unique format. XSL is made of two formalisms:

- XSL FO (Formatting Objects) that enables the description of edition formats and graphical outputs such as margins, fonts, blocks, lines, boxes, etc.
- XSLT (XSL Transform) that enables XML tree transformation, and generation of XML, HTML, and text files.

XSL FO will be used by software editors to build translation tools producing proprietary formats from XML such as PDF, Word, etc. On the contrary, XSLT is extremely useful for transformation in the world of XML and Web documents.

[1] http://www.w3.org/TR/xpath
[2] http://www.w3.org/XML/Linking
[3] http://www.w3.org/Style/XSL/

XSLT is a language that enables to specify transformation of XML documents to other XML (or XHTML) documents. It is for example possible to generate a table of contents of documents, adapt sorting of tables, dynamically build multiple HTML views to navigate a single XML document.

XSLT is a rule-based language where formatting rules are called templates. A template matches elements on which to apply a transformation pattern. The example in Figure 21 shows a template that applies to elements *chapter* that are in a *book* element, that extracts the *title* of such elements and that calls for the application of templates on the children of *chapter*:

```
1   <xsl:template match='book/chapter'>
2      <xsl:value-of select='title'/>
3      <xsl:apply-templates/>
4   </xsl:template>
```

Figure 21 Simple title extraction template

As shown here, the pattern of the template uses XPath. XSLT operators enable to access the values (e.g. `xsl:value-of`) and branching instructions are available (e.g. `xsl:if`, `xsl:apply-templates`, etc.). Templates are applied recursively on the XML document, by means of the `apply-templates` instruction (as shown in the above example) that finds templates matching children of the current node and apply them. There are also facilities for sorting and counting elements, importing stylesheets, defining variables and parameters, calling templates by name and passing parameters to templates.

Figure 22 presents a stylesheet extracting the name and the phone number from the document given in Figure 18. The output of this stylesheet is an HTML document given in Figure 23.

```
1   <xsl:template match="/">
2   <HTML>
3    <HEAD>
4     <TITLE>Phones</TITLE>
5    </HEAD>
6    <BODY>
7      <xsl:apply-templates />
8    </BODY>
9   </HTML>
10  </xsl:template>
11
12  <xsl:template match="contact_details">
13   <xsl:value-of select='name'>
14   <xsl:text>: </xsl:text>
15   <xsl:value-of select='phone'><BR/>
16  </xsl:template>
```

Figure 22 XSLT Sample

```
1   <HTML>
2    <HEAD>
3     <TITLE>Phones</TITLE>
4    </HEAD>
5    <BODY>
6       INRIA-Sophia: 04-92-38-77-00<BR>
7    </BODY>
8   </HTML>
```

Figure 23 HTML Result

The ability to dissociate structure content and presentation enables documents to be used and viewed in different ways by different human agents or software agents.

To parse and manipulate XML structures, the W3C standardised the DOM (Document Object Model). The DOM proposes a software structure created from a parsed XML document to be manipulated inside the program. It enables to create elements, get elements by tag name, access to attributes, navigate the tree structure, etc. The DOM API works on an explicit representation of the XML document as a forest. The Simple API for XML (SAX) is an event-driven API that operates

during parsing and runs associated call-backs. It is not a W3C recommendation, but it is a standard that emerged from the XML user community. Most XML parsers implement the SAX API.

Examples of languages defined in XML are:
- SMIL: Synchronised Multimedia Integration Language
- SVG: Scalable Vector Graphics, for describing two-dimensional graphics in XML
- XHTML: HTML with an XML syntax
- MathML: Mathematical Markup Language

Another example is the XML syntax of RDF(S) summarised in the next section.

3.3.2 RDF(S): Annotation approach

The W3C now proposes a framework for Web resource annotations with an associated XML syntax: RDF(S). This section merges and summarises the aspects of the specifications of RDF 1.0 and RDFS 1.0 that are relevant to the work presented here, including some slight modifications accounting for latest decisions of the W3C workgroups. The content given here makes extensive use of the W3C RDF Syntax recommendation [Lassila and Swick, 1999] [Manola and Miller, 2002] and RDFS Candidate Recommendation [Brickley and Guha, 2000] [Brickley and Guha, 2002]. The former explains the RDF data model used to represent metadata about the resources available on the Web and provides a syntax to serialise these descriptions in XML. The later defines a meta-model providing primitives to formalise classes and properties *i.e.,* ontological primitives which can be used to write RDF annotations. RDF(S) is the notation adopted here to denote the RDF & RDFS couple that are respectively detailed in the two following subsections.

3.3.2.1 Resource Description Framework: RDF datamodel

Resource Description Framework (RDF) provides "the necessary foundation and infrastructure to support the description and management of (Web) data." [Berners-Lee, 1999]. RDF is a foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web. RDF emphasises facilities to enable automated processing of Web resources *i.e.,* any entity addressable by a URI, and enables to describe it by writing statements about it.

The broad goal of RDF is to define a mechanism for describing resources that makes no assumptions about a particular application domain nor defines, *a priori*, the semantics of any application domain. RDF can be characterised as a simple frame system that does not specify a mechanism for reasoning; such mechanism must be built on top of this frame system. However the latest RDF Model theory [Hayes, 2002] defines some elementary inferences.

The RDF recommendation introduces a model for representing metadata as well as a syntax for encoding and transporting this metadata that uses XML. RDF and XML are complementary: RDF is a model of metadata and relies on XML to address by reference many of the encoding issues that transportation and file storage require. XML syntax is only one possible syntax for RDF and alternate ways to represent the same RDF data model may emerge.

The foundation of RDF is a model for representing named properties and property values. RDF properties are attributes of resources and in this sense correspond to traditional object-attribute-value triples. RDF properties also represent relationships between resources and therefore, an RDF model can resemble an entity-relationship diagram. In object-oriented design terminology, resources correspond to objects and properties correspond to instance variables with that difference that, as we shall see in RDFS, properties are defined outside the classes.

The basic data model thus consists of four object types:
- *Resources*: All things being described by RDF expressions are called resources. A resource may be an entire Web page such as the HTML document "http://www.w3.org/Overview.html" for example. A resource may be a part of a Web page e.g.: a specific HTML or XML element within the

document source. A resource may be an electronic document e.g.: an image, a zipped file, etc. A resource may be a whole collection of pages; e.g. an entire Web site. A resource may also be an object that is not directly accessible via the Web e.g.: a printed book. Resources are addressed by their URIs plus optional anchor ids. An anonymous resource expresses an existential quantification. Anything can have a URI; the extensibility of URIs allows the introduction of identifiers for any entity imaginable i.e., URIs are not limited to URLs. This is important because we can reference legacy resources and physical resources.

- *Properties*: A property is a specific aspect, characteristic, attribute, or relation used to describe a resource. Each property has a specific meaning, defines its permitted values, the types of resources it can describe, and its relationships with other properties. We shall see how the RDF Schema specification addresses the declaration of such characteristics.
- *Literal*: The most primitive value type represented in RDF is the Literal i.e., an arbitrary string of characters. The content of a literal is not interpreted by RDF itself and may contain additional XML markup. Literals are distinguished from Resources in that the RDF model does not permit (yet) literals to be the subject of a statement. In RDF terms, a literal may have content that is XML markup, but is not further evaluated by the RDF processor. Discussions on improving the data typing are currently fostered by the introduction of XML types and the consideration on a complete model theory.
- *Statements*: A specific resource together with a named property plus the value of that property for that resource is an RDF statement. These three individual parts of a statement are called, respectively, the subject, the predicate, and the object. The object of a statement (i.e., the property value) can be another resource or it can be a literal.

Thus the RDF data model is defined formally as follows:

1. There is a set called Resources.
2. There is a set called Literals.
3. There is a subset of Resources called Properties.
4. There is a set called Statements, each element of which is a triple of the form *(pred, sub, obj)*.

Where *pred* is a property (member of Properties), *sub* is a resource (member of Resources), and *obj* is either a resource or a literal (member of Literals). The triple is the quantum of knowledge representation in RDF.

We can view a set of statements (members of Statements) as a directed labelled graph: each resource and literal is a vertex; a triple *(predicate, subject, object)* is an arc from *subject* to *objec*t, labelled by *predicate* as shown in Figure 24. The triple 3D view was inspired by the excellent book "Gödel, Escher, Bach: An Eternal Golden Braid" by Douglas R. Hofstadter [Hofstadter, 1999].

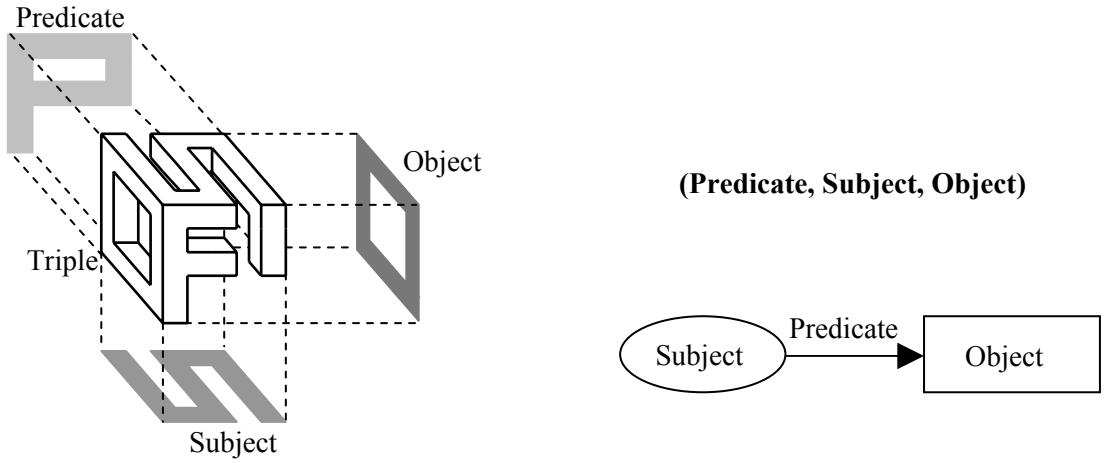
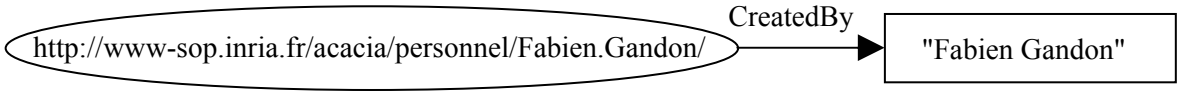


Figure 24 The RDF triple: Smallest statement and quantum of annotation knowledge

Using this model, the fact that "Fabien Gandon" created his Web Page is represented in Figure 25 using the graph notation, the triple notation and the XML associated syntax.

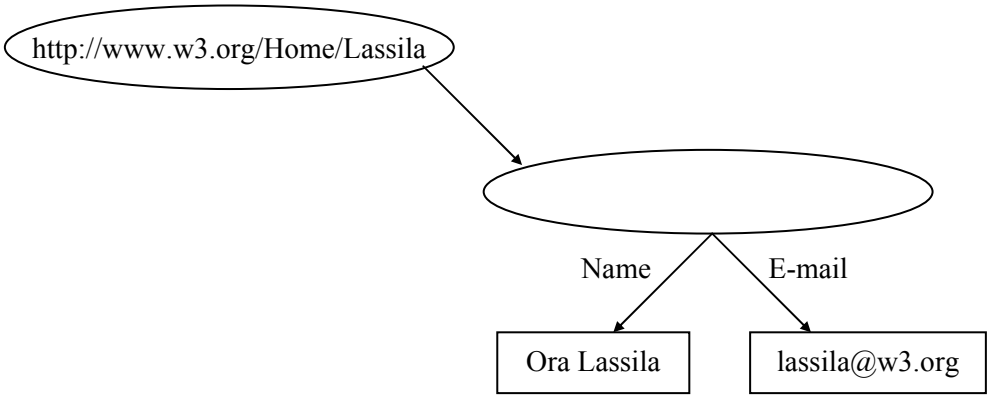
```
<?xml version="1.0"?>
  <rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:s="http://description.org/schema/">
   <rdf:Description rdf:about="http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/">
     <s:CreatedBy>Fabien Gandon</s:CreatedBy>
   </rdf:Description>
  </rdf:RDF>
```



{CreatedBy, http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/, "Fabien Gandon"}

Figure 25 Example of RDF Annotation

The RDF data model intrinsically only supports binary relations; that is, a statement specifies a relation between two resources. Illustrating the problem of granularity or conceptualisation we addressed before, the RDF recommendation gives the following example (Figure 26) where one wants to be more precise saying that "The individual whose name is Ora Lassila, and email is <lassila@w3.org>, is the creator of http://www.w3.org/Home/Lassila".




```
<rdf:RDF>
  <rdf:Description rdf:about="http://www.w3.org/Home/Lassila">
    <s:Creator>
      <rdf:Description>
        <v:Name>Ora Lassila</v:Name>
        <v:Email>lassila@w3.org</v:Email>
      </rdf:Description>
    </s:Creator>
  </rdf:Description>
</rdf:RDF>
```

Figure 26 Annotation Example from RDF specifications

In this example, the individual instance is existentially quantified since there is no URI to address it.

For typing, the RDF data model is completed by a property named *rdf:type*. It is defined to provide primitive typing. Thus, continuing the formal definition:

5. There is an element of Properties known as *rdf:type*.
6. Members of Statements of the form *(rdf:type, sub, obj)* must satisfy that *sub* and *obj* are members of Resources. RDFS will place additional restrictions on the use of *rdf:type*.

3.3.2.2 RDF Schema: RDFS meta-model

Resource description communities require the ability to express things about certain kinds of resources. For describing bibliographic resources, for example, descriptive concepts and attributes include "Book", "Author", "Title", etc. The meaning of this vocabulary is crucial to understanding the statements and establishing that the correct processing occurs as intended. It is vital that both the writer and the reader of a statement understand the same meaning for the symbolic ID used, such as "ReviewedBy", "Copyright", "Book", etc. or ambiguity and incoherence will result. Here we see the issues of ontologies reappear again. The data model described previously does not provide mechanisms for declaring these properties, nor does it provide any mechanism for defining the relationships between these properties and other resources. That is the role of RDF Schema (RDFS): to facilitate the definition of metadata by providing a class system much like many object-oriented programming and modelling systems.

The declaration of these properties (attributes) and their corresponding semantics are defined using RDFS. RDFS does not specify a vocabulary of descriptive elements such as "Book" or "Title". Instead, it specifies the mechanisms needed to define such elements *i.e.,* to define the classes of resources that may be used with, to restrict possible combinations of classes through relationships, and to detect violations of those restrictions. Thus RDFS defines a *schema specification language*.

RDF Schema provides a basic type system for use in RDF models. It defines resources and properties such as `rdfs:Class` and `rdfs:subClassOf` that are used in specifying application-specific schemas. Classes and Properties are organised in hierarchies, and offer extensibility through subclass refinement. A schema is also the place where definitions and restrictions of usage for properties are documented.

The typing system is specified in terms of the basic RDF data model - as resources and properties. Thus, the resources constituting this typing system become part of the RDF model of any description that uses them. The schema specification language is a declarative representation language influenced by ideas from knowledge representation as well as database schema specification languages and graph data models. The RDF Schema specification language is far less expressive, but far much simpler to implement, than full predicate calculus languages.

RDF Schemas might be contrasted with XML Document Type Definitions (DTDs) and XML Schemata. Unlike an XML DTD or Schemata, which gives specific constraints on the structure of an XML document, an RDF Schema provides information about the interpretation of the statements given in an RDF data model. While a DTD or an XML Schema can be used to validate the syntax of an RDF/XML expression, a syntactic schema alone is not sufficient for RDF purposes. RDF Schemas may also specify constraints that should be followed by these data models. As RDF uses XML for its interchange encoding, the work on data typing in XML itself will be the foundation for such a capability.

Resources may be instances of one or more classes; this is indicated with the `rdf:type` property. The RDF Schema class hierarchy is similar to the type systems of object-oriented languages. However, RDF differs from many such systems in that instead of defining a class in terms of the properties its instances may have, an RDF schema will define properties in terms of the classes of resource to which they apply; it could be said to be a graph-oriented view. The signature of a property is specified by the `rdfs:domain` and `rdfs:range` constraints. For example, we could define the "Title" property to have a domain "Book" and a range "Literal". One benefit of the RDF property-centric approach is that it is very easy for anyone to say anything about existing resources, and add new properties to existing schemas, which is one of the architectural principles of the Web.

Properties can only be binary. Initially, the signature of a property consisted of a unique range type constraint, but could have several domain type constraints (the semantics being not clear). Lately, discussions on the evolution of RDFS tend to consider that multiple range and domain will be allowed and that the attached semantics will be the conjunction of the types *i.e.,* the instance must be of all the given types (which is possible, especially in RDF where multi-instantiation is allowed).

A regrettable restriction is that it is not possible to add constraints to restrict the range of a property when it is applied to a subclass *i.e.,* it is impossible to overwrite the signature for sub-classes of the domain and the range while keeping the same ID for the property.

Figure 27 uses a "nodes and arcs" graph representation of the RDF data model. RDFS metamodel is also expressed in RDF since everything is a resource. `rdfs:Resource` is the top class in the metamodel. Two particular classes `rdfs:Class` and `rdf:Property` are the classes that gather all the classes and properties. `rdfs:subClassOf`, `rdfs:subPropertyOf` and `rdf:type` are three instances of `rdf:Property`. Initially `rdfs:subPropertyOf` and `rdfs:subClassOf` properties should not form loops, but this restriction was lifted. If one class is a sub-class of another, then there is an `rdfs:subClassOf` arc from the node representing the first class to the node representing the second. Similarly, if a resource is an instance of a class, then there is an `rdf:type` arc from the resource to the node representing the class. The figure only shows the arcs to the most tightly encompassing class, and rely on the transitivity of the `rdfs:subClassOf` relation to provide the remainder. The primitives shown in bold are the most important ones and others are not shown because they will not be used here.

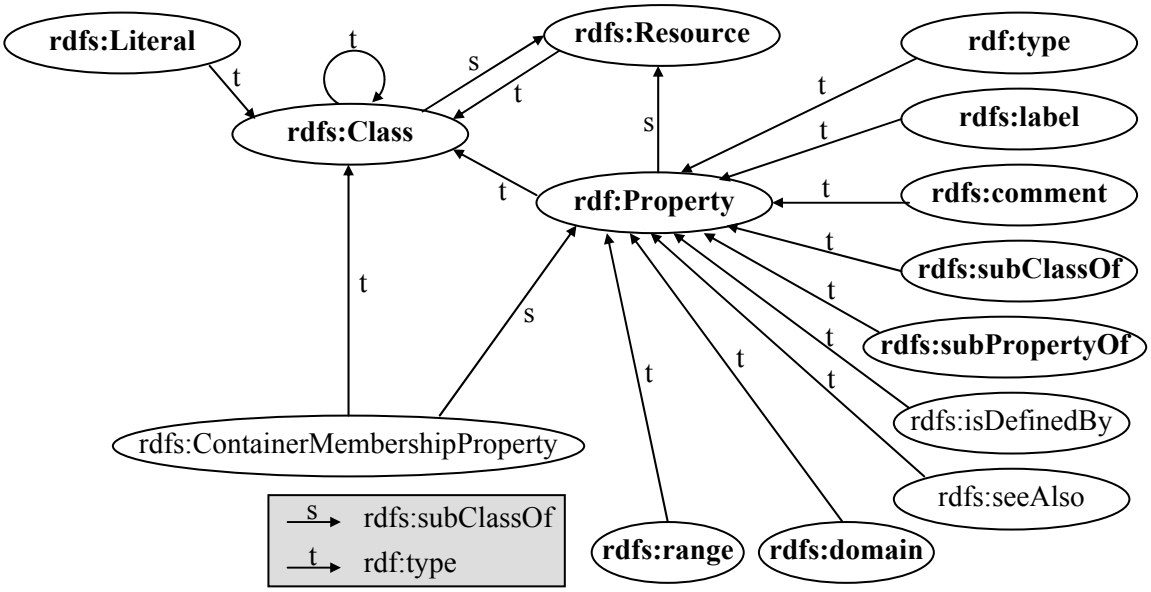


Figure 27 RDF Data model latest draft to date

The meta-model is defined in a namespace informally called 'rdfs' here, and identified by the URI reference http://www.w3.org/2000/01/rdf-schema#.

There are two loops in the meta-model:
- A mutual reference: Class is a kind of Resource (sub type of) and Resource is a Class (type).
- A self-reference: class is a class (type).

This is comparable to meta-models found for instance in LISP/CLOS. It introduces a terminal point in the graph enabling the model to rely on itself. It may seem contradictory with the research for non recursive definitions as explained in the ontology section. However, for me, it is only the expression of the limit of the scope of the meta- model *i.e.,* any further abstract description would be useless. As a result the meta-model can entirely express itself in the triple model of RDF.

RDF Schema provides a mechanism for describing properties signature constraints, but does not specify whether or how an application must process the constraint information. For example, while

an RDF schema can assert that an author property is used to indicate resources that are members of the class Person, it does not specify whether or how an application should act in processing that class information. Different applications may use these constraints in different ways - e.g. a validator will look for errors, an interactive editor might suggest legal values, and a reasoning application might infer the class and then announce all inconsistencies.

RDF schemas can express constraints that relate vocabulary items from multiple independently developed schemas. Since URI references are used to identify classes and properties, it is possible to create new properties which domain or range constraints reference classes defined in other namespaces.

The following properties are provided to support simple documentation and user-interface related annotations within RDF schemas:
- `rdfs:comment`: This is used to provide a human-readable description of a resource.
- `rdfs:label`: This is used to provide a human-readable version of a resource name.

Multilingual documentation of schemas is supported at the syntactic level through use of the `xml:lang` language tagging facility. Since RDF schemas are expressed within the RDF data model, vocabularies defined in other namespaces may be used to provide richer documentation. These properties can be used to capture natural language terms and definitions attached to concepts intensions.

An example summarising the use of RDF(S) to formalise a hierarchy of concepts and relations is given in Figure 28 and an annotation is given in Figure 29.

In order to create a schema slightly different from an existing one, one can just provide incremental modifications to some existing base schema. To avoid confusion between independent and possibly conflicting definitions having the same ID, the XML syntax of RDF uses the XML namespaces to tie a specific use of a concept ID to the URI of the schema giving its intended definition; the URI used as the namespace of a schema provides a unique identifier for this schema. Since each RDF schema has its own unchanging URI, this one can be used to construct unique URI references for the resources defined in a schema. This is achieved by combining the local identifier for a resource with the URI associated with that schema namespace. Thus each predicate used in a statement must be identified with exactly one namespace, or schema. However, a Description element may contain statements with predicates from many schemas. Since changing the logical structure of a schema risks breaking other RDF models which depend on that schema, the specification recommends that a new namespace URI should be declared whenever an RDF schema is changed. In effect, changing the RDF statements which constitute a schema creates a new one; new schema namespaces should have their own URI to avoid ambiguity. Since an RDF Schema URI unambiguously identifies a single version of a schema, software that uses or manages RDF (e.g. caches) should be able to safely store copies of RDF schema models for an indefinite period.

Through the sharability of schemas, RDF supports the reusability of metadata definitions. Due to RDF's incremental extensibility, software should be able to trace the origins of schemata they are unfamiliar with back to known schemata and perform meaningful actions on metadata they were not originally designed to process. The sharability and extensibility of RDF also allows metadata authors to use multiple inheritance to "mix" definitions, to provide multiple views to their data, leveraging work done by others. In addition, it is possible to create RDF instance data based on multiple schemata from multiple sources (*i.e.*, "interleaving" different types of metadata).

Additional points such as reified statements and containers are not addressed here because they are irrelevant to this work.

```
1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <rdf:RDF xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns="http://www.w3.org/TR/REC-html40" xmlns:cos="http://www.inria.fr/acacia/corese#">
3   <!-- O'CoMMA Ontology version 4.4 -->
    ...
4   <rdfs:Class rdf:ID="Document">
5    <rdfs:subClassOf rdf:resource="#Entity"/>
6    <rdfs:subClassOf rdf:resource="#EntityConcerningATopic"/>
7    <rdfs:subClassOf rdf:resource="#NumberableEntity"/>
8    <rdfs:comment xml:lang="en">Entity including elements serving as a representation
       of thinking.</rdfs:comment>
9    <rdfs:comment xml:lang="fr">Entite comprenant des elements de representation de la
       pensee.</rdfs:comment>
10   <rdfs:label xml:lang="en">document</rdfs:label>
11   <rdfs:label xml:lang="fr">document</rdfs:label>
12  </rdfs:Class>
13  ...
14  <rdfs:Class rdf:ID="Article">
15   <rdfs:subClassOf rdf:resource="#Document"/>
16   <rdfs:subClassOf rdf:resource="#ExtractedDocument"/>
17   <rdfs:comment xml:lang="en">Document corresponding to a piece of writing on a
        particular subject and which purpose is to fully realize a particular objective
        in a relatively concise form e.g.: demonstrate something.</rdfs:comment>
18   <rdfs:comment xml:lang="fr">Document correspondant un texte sur un sujet particulier et
        qui a pour but de realiser un objectif particulier sous une forme relativement
        concise, par exemple : demontrer quelque chose.</rdfs:comment>
19   <rdfs:label xml:lang="en">article</rdfs:label>
20   <rdfs:label xml:lang="fr">article</rdfs:label>
21  </rdfs:Class>
    ...
22  <rdf:Property rdf:ID="HasForISBN">
23   <rdfs:subPropertyOf rdf:resource="#HasNumber"/>
24   <rdfs:range rdf:resource="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#Literal"/>
25   <rdfs:domain rdf:resource="#Book"/>
26   <rdfs:comment xml:lang="en">The International Standard Book Number (ISBN) is a system
      of numerical identification for books, pamphlets, educational kits, microforms, CD-ROM
      and braille publications.</rdfs:comment>
27   <rdfs:comment xml:lang="fr">Le numero standard international de livre (ISBN) est un
      systeme d identification numerique pour les livres, les brochures,les kits educatifs,
      les microformes, les CD-ROM et les publications en braille.</rdfs:comment>
28   <rdfs:label xml:lang="en">ISBN</rdfs:label>
29   <rdfs:label xml:lang="en">International Standard Book Number</rdfs:label>
30   <rdfs:label xml:lang="fr">ISBN</rdfs:label>
31   <rdfs:label xml:lang="fr">International Standard Book Number</rdfs:label>
32   <rdfs:label xml:lang="fr">numero standard international de livre</rdfs:label>
33  </rdf:Property>
    ...
34  </rdf:RDF>
```

Figure 28 Extract of an RDF schema.

```
1   <!DOCTYPE rdf:RDF [ <!ENTITY nsCoMMA "http://www.inria.fr/acacia/comma#" > ]>
2   <rdf:RDF
3     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
5     xmlns:CoMMA="&nsCoMMA;">
6     <CoMMA:Article rdf:about="http://www-
    sop.inria.fr/acacia/personnel/Fabien.Gandon/research/pakm2000/pakm2000.pdf">
7       <CoMMA:Title>Presentation of advantages of XML and MAS in KM</CoMMA:Title>
8       <CoMMA:CreatedBy>
9        <CoMMA:Person rdf:about="http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/" />
10      </CoMMA:CreatedBy>
11      <CoMMA:HasForPerceptionMode>
12        <CoMMA:VisualPerceptionMode  rdf:about="&nsCoMMA;VisualPerceptionMode"/>
13      </CoMMA:HasForPerceptionMode>
14      <CoMMA:HasForRepresentationSystem>
15        <CoMMA:English rdf:about="&nsCoMMA;English"/>
16      </CoMMA:HasForRepresentationSystem>
17      <CoMMA:Target>
18        <CoMMA:OrganizationalEntity>
19          <CoMMA:IsInterestedBy>
20           <CoMMA:KnowledgeManagementTopic rdf:about="&nsCoMMA;KnowledgeManagementTopic"/>
21          </CoMMA:IsInterestedBy>
22          <CoMMA:IsInterestedBy>
23           <CoMMA:MultiAgentSystemTopic rdf:about="&nsCoMMA;MultiAgentSystemTopic"/>
24          </CoMMA:IsInterestedBy>
25          <CoMMA:IsInterestedBy>
26           <CoMMA:XMLTopic rdf:about="&nsCoMMA;XMLTopic"/>
27          </CoMMA:IsInterestedBy>
28        </CoMMA:OrganizationalEntity>
29      </CoMMA:Target>
30    </CoMMA:Article>
31  </rdf:RDF>
```

Figure 29 An RDF annotation describing an article

3.4 Summary and perspectives

To summarise the current vision of the Semantic Web, Tim Berners-Lee uses in his presentations the Semantic Web Stack (or Semantic Web Cake) as reproduced in Figure 30.

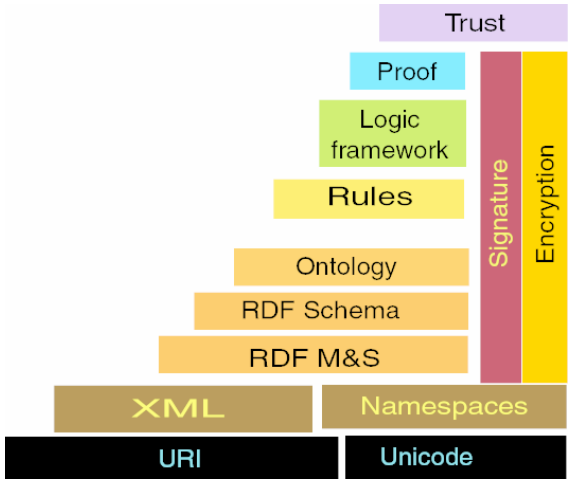


Figure 30 Semantic web stack viewed by Tim Berners-Lee

The lowest layers ensure syntactic interoperability: Unicode is a universal text encoding standard and URIs provide a universal addressing standard. XML provides a standard to describe the structure of documents in an extensible fashion, to create instances of documents, to manipulate them, to link them and to validate them. Namespaces provide a standard to identify names of tags used for structuring XML documents.

However the syntactic interoperability is not enough to allow a software to 'understand' the content and manipulate it in a meaningful and useful way. The syntax allow us to declare a tag `<article>`, but it has no more meaning for the machine than the tag `<jH6lZa>` has meaning for us. The systems still work at the symbolic level while we want them to address semantic problems.

RDF Model and Syntax and RDF Schema Layers lift us to the point where we can describe taxonomies of concepts and properties (with their signatures). The XML syntax of RDF place it above the previous layers thus benefiting of the syntactic interoperability they ensure. These additional layers lay the first foundations of a semantic interoperability.

RDF(S), like it was the case for XML, requires supporting tools. Such implementations can use available XML tools (XML parsers, stylesheets and associated engines to manipulate RDF(S), etc.), but these tools are not sufficient. RDF(S) parsers are required to provide RDF(S)-level events and structure; they can be built above XML parsers, but XML events are too low level for RDF(S) manipulation. Moreover, RDF(S) is used to annotate resources to improve automatic processing; the first one being searching for relevant resources. Initiatives like XML Query are too low level; they are at the syntactic level and, for instance, do not take into account the existence of RDF Schema. So, in the first following sub-part, I shall mention some existing implementations.

Moreover, we know that ontology formalisation can require additional expressiveness that RDF(S) minimalist approach does not provide. RDF(S) is simple enough to be widely accepted and used and start to improve current Web facilities; it is a necessary ground and as we shall see it already enables to do interesting work. However the additional expressiveness needed to improve granularity of ontologies has to be provided with additional layers above, especially rules and logics.

Thus the approach followed is quite wise: it is a progressive addition of additional expressiveness layers. This promotes a progressive development and deployment of demonstrator prototypes participating in prototype lifecycle development. It allows anyone to chose the degree of expressiveness required for an application thus stopping at the right layer. In that perspective, I shall mention in the second following sub-part, some extension initiatives.

3.4.1 Some implementations of the RDF(S) framework

3.4.1.1 ICS-FORTH RDFSuite

ICS-FORTH RDF Suite[1] provides open source scalable tools for the Semantic Web. It is the result of a work on how to store an manipulate RDF(S) in a relational database to improve the speed and scalability of processing. However, there is no additional expressiveness and inference capabilities built on top of RDF. This suite includes

- The Validating RDF Parser (VRP): an RDF Parser supporting semantic validation of both resource descriptions and schemas.
- The RDF Schema Specific DataBase (RSSDB): RDF store using schema knowledge to automatically generate an Object-Relational (SQL3) representation of RDF metadata and load resource descriptions.
- The RDF Query Language (RQL): Declarative Language for uniformly querying RDF schemas and resource descriptions.

RQL is based on OQL (ODMG standard for querying object-oriented databases). It enables to query both the annotations in RDF and the schema in RDFS. This approach is natural in RDF since classes and properties are considered to be resources and can be retrieved in the same way as annotation resources. Querying an unknown schema allows to discover it in order to use it afterwards to query the annotations. RQL supports generalised path expressions featuring variables on both labels for nodes (*i.e.*, classes) and edges (*i.e.*, properties); the expression can make use of regular expression to express constraints.

The RQL Interpreter is developed in C++ and consists of three modules:

- the Parser, analysing the syntax of queries;
- the Graph Constructor, capturing the semantics of queries in terms of typing and interdependencies of involved expressions;
- the Evaluation Engine, accessing RDF descriptions from the underlying database via SQL3 queries.

3.4.1.2 Jena

Jena is a Java RDF API and toolkit part of the Hewlett-Packard Semantic Web activity[2]. The latest release of the Jena to date integrates a number of new components, some of which are also available separately:

- ARP parser and RDFFilter compliant with latest working group recommendations.
- API for manipulating RDF models including statement and resource-centric methods using cascading calls for easy object-oriented use.
- built in support for RDF containers - bag, alt and seq.
- RDF/XML writer.
- RDQL query language with the associated query engine and command line support for exploring data sets.
- support for storing DAML ontologies in a model.
- persistent storage module based on Berkeley DB and support for persisting Jena models in relational databases.
- open architecture supporting other storage implementations.

[1] http://www.ics.forth.gr/proj/isst/RDF/
[2] http://www.hpl.hp.com/semweb/index.html

RDQL is an SQL-like query language for RDF derived from SquishQL. It treats RDF as data and provides query with triple patterns and constraints over a single RDF model. The target usage is for scripting and for experimentation in information modelling languages.

Jena is available under an open source, BSD-like license.

3.4.1.3 SiLRI and TRIPLE

SiLRI[1] stands for Simple Logic-based RDF Interpreter [Decker *et al.*, 1998]. It is a main-memory logic-based inference engine implemented in Java. It implements a major part of Frame-Logic and has support for RDF. The Inference Engine uses SiRPAC parser to load RDF, but queries and rules must be submitted using F-Logic expressions. The system is also able to handle formulas and data in a datalog like notation. SiLRI offers a light weight component that can be integrated in other Java applications. The inference engine is available under a GNU public license

SiLRI will be commercialised by Ontoprise GmbH. The open-source successor of SiLRI is TRIPLE. TRIPLE[2] is an RDF query, inference, and transformation language. Instead of having a built-in semantics for RDF Schema as many other RDF query languages have, TRIPLE allows the users to chose the semantics of a language on top of RDF (like RDF Schema, Topic Maps, UML, etc.) to be defined with rules. For languages where this is not easily possible (e.g. DAML+OIL), access to external programs (like description logics classifiers) is provided. As a result, TRIPLE allows RDF reasoning and transformation under several different semantics.

3.4.1.4 CORESE

Conceptual graphs are well adapted to process RDF, because both formalisms consider properties as first class entities. This is the approach taken for CORESE[3] (Conceptual Resource Search Engine) [Corby *et al.*, 2000] [Corby and Faron-Zucker, 2002] used in my Ph.D. It will be detailed in chapter 6. It is a main-memory RDF(S) engine written in Java and it offers good performances while being relatively small in size. It makes it a good candidate as light-weight component for portable software in Java, easy to integrate within other Java software using its API. Its query language relies on RDF and RDFS patterns using variables, operators and regular expressions. The engine applies a customised version of the projection operator in Conceptual Graphs in order to provide all the relevant answers taking into account the hierarchies of the schema. An RDF-based rule language allows the users to complement the ontology with inference rules.

3.4.1.5 Karlsruhe Ontology (KAON) Tool Suite

KAON[4] is an open-source ontology management infrastructure allowing ontology creation and management. This tool suite has been developed in the context of the KAON Semantic Web infrastructure and provides an API for manipulating ontologies and instances with several storage mechanisms. KAON especially works with RDF Schema and DAML+OIL ontologies. KAON-API only supports some simple queries for browsing the ontology (e.g., get all classes or get all resources of a class) and does not allow specifying a filter or an expression path within the query. However KAON proposes several modules:

- Engineering Server: a storage mechanism for ontologies based on relational databases, suitable for use during ontology engineering.
- Kaon Portal: a tool for generating multi-lingual, ontology-based Web portals.
- OI-Modeler: a tool for ontology creation and evolution.
- Text-To-Onto: supports semi-automatic creation of ontologies by applying text mining algorithms.
- Rdf Crawler: a simple tool for syndication of RDF-based content from a set of Internet pages.

[1] http://www.aifb.uni-karlsruhe.de/~sde/rdf/
[2] http://triple.semanticweb.org/
[3] http://www-sop.inria.fr/acacia/soft/corese.html
[4] http://kaon.semanticweb.org/

3.4.1.6 Metalog

Metalog[1] [Marchiori and Saarela, 1998] uses techniques of logic programming to query and reason on RDF metadata. Querying language dedicated to the writing of inference rules and queries in English-like syntax. Metalog focuses on providing a logical view of metadata. Metalog provides a logical interpretation of RDF data into logic programming; this logical layer enables users to express logical expressions with operators to reason with RDF statements. The RDF Metalog schema extends RDF with the full power of first-order predicate calculus. To process and effectively reason with the translated logical formula, Metalog deals with a subset of first-order predicate calculus: Horn clauses. Metalog is provided with an interface to write structured data and reasoning rules using controlled natural-language.

3.4.1.7 Sesame

Sesame[2] is an RDF Schema-based Repository and Querying facility. Originally, it was developed by Aidministrator as a research prototype and one of the key deliverables in the European IST project On-To-Knowledge (EU-IST-1999-10132). Sesame is being now further developed by Aidministrator, OntoText and the NLnet Foundation as an Open Source tool under the terms of the GNU Lesser General Public License (LGPL).

There are four modules in the Sesame system:
- A data administration module for adding and removing data.
- An RDF Export module that can be used to export (parts of) the data in a repository as an RDF document.
- An RQL query engine that can be used to evaluate RQL queries for querying schemas (see ICS Forth).
- An RDQL query engine that can be used to evaluate RDQL queries (see Jena).

Two other modules are developed by OntoText:
- A security module for more fine-grained security mechanisms.
- A versioning module for ontology versioning.

A generic API for RDF (Schema) repositories known as RDF SAIL (Storage And Inference Layer) is used to abstract from the actual storage device. This API has methods for storing, removing and querying RDF in/from a repository. Implementations of this API are used by Sesame's functional modules, but can also be used by stand-alone tools that need some kind of repository for storing their RDF. It is possible to build a SAIL implementation on top of any kind of storage; be it a database, a file, or a peer-to-peer network. Current implementations are based on relational databases. This point is interesting for wrappers development.

Communication with the functional modules from outside the Sesame system is handled by protocol handlers. At this moment there is only one protocol handler that handles communication over HTTP, but another one for SOAP is envisaged. Other protocol handlers can be added.

3.4.1.8 Protégé-2000

Protégé[3] is a tool which allows the user to: construct a domain ontology; customise knowledge-acquisition forms; enter domain knowledge. It is also a platform which can be extended with graphical widgets for tables, diagrams, animation components to access other knowledge-based systems embedded applications. Finally it is a library which other applications can use to access and display knowledge bases. The knowledge model of Protégé-2000 is OKBC-compatible and can export to RDF(S). One of the major advantages of the Protégé is that the it is constructed in an open, modular fashion . Its component-based architecture enables system builders to add new functionality

[1] http://www.w3.org/RDF/Metalog/
[2] http://sesame.aidministrator.nl/
[3] http://protege.stanford.edu

by creating appropriate plug-ins. The Protégé plug-in library contains plug-ins for graphical visualization of knowledge bases, inference-engine for verification of constraints in first-order logic, acquisition of information from remote sources such as UMLS and WordNet, semi-automatic ontology merging, and many others.

3.4.1.9 WebODE

WebODE[1] is an ontological engineering workbench that provides ontology related services, and tries to cover and give support to most of the activities involved in the ontology development process. The platform has been built using a three-tier model, using an application server basis. It provides services for ontology edition, integration, administration and user collaboration. It is based on an expressive internal knowledge model but it provides an API for ontology access and integration with other systems, as well as translators that import and export ontologies from and to markup languages (DAML+OIL, OIL, RDF(S), XML) and other traditional ontology languages.

3.4.1.10 Mozilla RDF

Finally, and to show that RDF(S) is coming closer to end-users, I mention the Mozilla RDF action[2] to provide a component-based API in Mozilla fostering the use of RDF data model as a mechanism for integrating and organising Internet resources. It also includes an umbrella project for experimental work investigating the integration of logic/inference capabilities into the Mozilla application environment.

[1] http://delicias.dia.fi.upm.es/webODE/index.html
[2] http://www.mozilla.org/rdf/doc/

3.4.2 Some extension initiatives

In the previous part, we saw that some existing implementations already started to introduce logical layers or rules as extensions to RDF(S). Therefore, there is a need to provide recommendations for the upper layers. I briefly survey here some of the extensions that were proposed to date.

3.4.2.1 DAML-ONT

DAML-ONT[1] was an initiative part of the DARPA Agent Markup Language to build the semantic web, focusing on the eventual creation of a web logic language. DAML-ONT was a draft language for ontology description. The language did not include any specification of explicit inference rules. The language extends RDFS with new primitives:

- `disjointFrom`, expresses that two classes are disjoint.
- `complementOf`, expresses an anonymous class as the complement of another.
- `disjointUnionOf`, expresses the partition of a class into several subclasses.
- `oneOf` enables to state all the possible elements of a class corresponding to an enumerated set.
- `intersectionOf`, expresses the fact that a class is the intersection of several classes.
- `TransitiveProperty`, expresses the transitive nature of a property.
- `inverseOf`, expresses the inverse property of a property.
- `equivalentTo`, `sameClassAs`, `samePropertyAs`, express equivalence between two classes or between two properties.
- `cardinality` and `maxCardinality`, that express exact and maximal cardinality of a property.
- `restrictedBy`, `Restriction`, `onProperty`, `toClass`, `toValue`, express that the range of a property is restricted when applied to a subclass of the domain.
- `UnambiguousProperty`, expresses that a property is an identity condition *i.e.,* can be used to differentiate or assimilate two individuals.

A group of researchers from MIT's Laboratory for Informatics had primary responsibility for developing this language core. This was then extended by a number of people including representatives from the OIL effort, the SHOE project, the KIF work, and DAML contractors. The successor of DAML-ONT is DAML+OIL.

3.4.2.2 OIL and DAML+OIL

OIL (Ontology Inference Layer) [Fensel *et al.*, 2000] is a formalism based on RDFS and extends it using description logics (concepts and roles), frame languages (properties are second-class entities declared inside the definition of a class). It was part of On-To-Knowledge, a European Project dedicated to knowledge management in large intranets by modelling and annotating the semantics of information using ontologies.

OIL aims at providing the greatest expressiveness possible, but with complete algorithms; thus it comes with a sound and complete reasoner called FACT, inherited from description logics. However, and as far as I know, this engine only reasons on the ontological level and not yet on the annotation level; therefore, it is not clear if it can be used for querying.

OIL presents a layered approach to a standard ontology language. Each additional layer adds functionality and complexity to the previous layer. This is done such that agents (humans or machines) who can only process a lower layer can still partially understand ontologies that are expressed in any of the higher layers.

[1] http://www.daml.org/2000/10/daml-ont.html

The different layers are:
- *Core OIL*: it coincides largely with RDF Schema. This means that even simple RDF Schema agents are able to process the OIL ontologies, and pick up as much of their meaning as possible.
- *Standard OIL*: it is a language intended to capture the mainstream modelling primitives for the description of an ontology.
- *Instance OIL*: it is the integration of individuals based on database capability. It has the same schema as Standard OIL, but adds instances described in RDF.
- *Heavy OIL*: it will include additional representational (and reasoning) capabilities.

Among the different primitives introduced in Standard OIL and extending RDFS are:
- *Class definitions*: a class can be primitive or defined. The definition of a defined class consists of the list of its super classes and a class expression being either a class name, a slot constraint or a boolean combination of class expressions. The description logics influence gave this ability to define concepts; it is interesting especially for classification and implicit knowledge discovery. The frame influence led classes to have internal properties called attributes. Relations can still be defined and organised in hierarchies as classes.
- *Property definitions*: likewise, it was influenced by slot definition in description logics that associates a slot name with a slot description that specifies constraints applying to the slot relations; mainly cardinality and value restrictions.
- *Class expressions*: where classes are needed, they can be replaced by a class expression, which is a set of slot constraints or of boolean combination of other class expressions. The boolean operators supported are OR, AND and NOT. For instance, a class A can be declared to be a subclass of the following class expression: (B OR C) AND NOT D.
- *Restriction on relation properties*: a subclass can have more specific properties values than its superclass.
- *Disjoint classes*: the disjunction of classes is expressed by an additional primitive.
- *Algebraic properties of relations*: a fixed number of algebraic properties can then be expressed: transitivity, symmetry, and inverse slot.

DAML+OIL[1] is the evolution of DAML-ONT compatible with OIL primitives.

3.4.2.3 DRDF(S)

DRDF(S) stands for Defined Resource Description Framework Schema [Delteil *et al.*, 2001]. It is an extension of RDF(S) dedicated to ontology representation on the Semantic Web. It was designed in the European IST project CoMMA. It enables the representation of axioms, class and property definitions in ontologies.

More generally, DRDF(S) provides a way to represent contextual knowledge on the Web. RDF(S) extensions offered by DRDF(S) are based on features of the Conceptual Graphs model to provide further representation capabilities

DRDF(S) enables:
- *Context definition*: DRDF(S) extends RDF with a notion of context to express the clustering of statements much more easily than RDF statements and containers. A context identifies a subgraph of the whole RDF graph, so that a triple can be stated inside of a special context. A context in DRDF(S) is the counterpart of a graph in Conceptual Graph. By introducing contexts in RDF, the authors proposed a very general mechanism for further extensions.
- *Class definition*: it was inspired by concept type definition in conceptual graphs. A class definition is a monadic abstraction, *i.e.,* a context having one resource of type Variable considered as formal parameter.

[1] http://www.daml.org/2000/12/daml+oil-index

- *Property definition*: it was inspired by relation type definition in conceptual graphs. A property definition is a diadic abstraction, *i.e.,* a context having two resources of type Variable considered as formal parameters.
- *Axiom definition*: it was inspired by graph rules in the Conceptual Graph model. Here again, an axiom definition relies on context definition: an axiom is a couple of lambda abstractions, *i.e.,* two contexts representing the hypothesis and the conclusion.
- *Extended existential quantification and co-reference*: We saw that RDF data model allows a limited form of existential quantification through anonymous resources. The introduction of the referent property of the Conceptual Graph model provides the RDF model with a general mechanism for existential quantification handling. An existential quantification is represented by an anonymous resource described by a referent property whose value is an instance of Variable. The scope of a variable is the context it belongs to.

3.4.2.4 OWL

OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. It is derived from the DAML+OIL Web Ontology Language and it is being designed by the W3C Web Ontology Working Group[1].

OWL is still under discussion, but it should extend RDFS with the following aspects:
- *Data typing*: OWL divides the universe into two disjoint parts. The first one consists of the values that belong to XML Schema datatypes (replaces Literals in RDF(S)). The second part consists of (individual) objects considered to be members of classes described (resources in RDF(S)).
- *Class equivalence*: `owl:sameClassAs` or `owl:equivalentTo` applied to classes express their equivalence
- *Property equivalence*: `owl:samePropertyAs` or `owl:equivalentTo` applied to properties express their equivalence.
- *Instance equivalence*: `owl:sameIndividualAs` states that two objects are the same while `owl:differentIndividualFrom` states that two objects are different.
- *Class expression*: class expression either refers to a named class, namely the class that is identified by the URI, or implicitly defines an anonymous class by an enumeration (`owl:oneOf`), a property-restriction, or a boolean combination of class expressions (`owl:intersectionOf`, `owl:unionOf`, `owl:complementOf`).
- *Property restrictions*: using the primitive `owl:Restriction`, value restrictions (`owl:allValuesFrom`, `owl:hasValue`, `owl:someValuesFrom`) and cardinality restriction (`owl:cardinality`, `owl:maxCardinality`, `owl:minCardinality`) can be stated.
- *Properties of properties*: `owl:inverseOf` indicates the inverse property. Algebraic properties are given by `owl:TransitiveProperty` and `owl:SymmetricProperty`. Unique values properties are given by `owl:FunctionalProperty` and `owl:InverseFunctionalProperty`

While it is widely appreciated that all of the features in knowledge modelling languages are important to some users, the working group of W3C also understood that languages as expressive may be daunting to some groups who are trying to support a tool suite for the entire language.

In order to provide a target that is approachable to a wider audience, a smaller language has been defined, referred to as OWL Lite[2]. It attempts to capture many of the commonly used features of OWL and also attempts to describe a useful language that provides more than RDFS with the goal of adding functionality that is important in order to support web applications. OWL Lite is basically a compatible subset of OWL.

[1] http://www.w3.org/2001/sw/
[2] http://www.w3.org/TR/2002/WD-owl-features-20020729/

3.5 Remarks and conclusions

Layer-based approach more or less followed in the different current trends is a good thing because even if above RDF(S) nothing is stable to date, one can start building, testing and using products for lower layers. The minimalist view of RDF(S) allows to refine the foundations and start spreading the ideas and the basics of the semantic Web.

The idea that the Web could be turned into a huge knowledge based named "Semantic Web" lead researchers of the knowledge modelling community to see in the "Semantic Web" the killer application field for their researches. While this could be the case there is a danger of forgetting the particular characteristics of the (semantic) web paradigm that motivated RDF(S) and just recycling well-known knowledge representation formalisms with an XML syntax.

The World Wide Web is ultimately managed and used by humans; mankind created the Web in its own image: a vast number of individuals distributed over large spaces, with conflicting opinions, with different points of view, with varied interests, with competing or orthogonal goals and means, with uncertain degrees of truthfulness, etc.

A semantic web formalism layer must clearly acknowledge that:
- *statements are claims*: a schema or an annotation on a semantic web is an individual conceptualisation or opinion made public; nothing more and nothing less. Claim context is important and provides a basic ground for truth management.
- *statements are distributed*: a schema or an annotation can be completed anywhere else, the notion of a comfortable centralised base no longer holds. If something cannot be proved, it may not be because it is false, it may be because the missing knowledge is somewhere out there on the open Web in a place or form we are not aware of.
- *statements are numerous*: classic knowledge engineering formalisms are known for their high computational complexity. Knowledge-based inferences quickly become NP-complete procedures, as soon as the language becomes sufficiently expressive and full first-order logic is even undecidable.

One could claim that the minimalist approach of RDF(S) to limit the expressiveness of the language is not viable. Yet the number of specification, implementation and deployment problems attest for the wisdom of the choice. Many open issues have been raised by RDF(S) and foster healthy discussions in the working groups.

In parallel, another trend can be followed and in fact, was followed in the history of the Internet and the Web: the progressive development of solutions at smaller scales and their eventual interconnection. Progressive organisation-wide solutions should address the above characteristics of the semantic web and propose software architecture to manage and exploit this special type of distributed knowledge; this distributed formal knowledge could very well be the playground of distributed artificial intelligence.

4 Distributed artificial intelligence

"The question of whether computers can think is like the question of whether submarines can swim."
"Computer science is not about computers any more than astronomy is about telescopes"
— EW Dijkstra

Artificial Intelligence is both a research field and its result: it is a branch of informatics concerned with the research on artificial systems performing activities that are normally thought to require intelligence such as reasoning [Haton *et al.*, 1991]; such a system is said to be an artificial intelligence. There exists a continuum between two extreme paradigms of artificial intelligence which spans from, on the one hand, a connectionism perspective, based on the simulation of elementary natural actors and the fostering of a collective intelligent behaviour emergence through interaction, to, on the other hand, symbolicism perspective, based on the explicit formal development of symbolic systems and associated interpretations to represent formal knowledge and simulate meaningful operations on it.

Although the connectionism perspective intrinsically has a collective dimension, both approaches led to monolithic systems (a neuronal network classifier, an expert system, etc.). In our physical world, situatedness, locality and distribution are parts of an unavoidable reality. Distributed artificial intelligence tries to understand the principles underlying the behaviour of multiple artificial intelligence entities, called agents, and their interactions in order to produce a meaningful overall multi-agent system behaviour. Therefore, study of such systems include and goes beyond the study of individual artificial intelligence, to consider the issues of artificial societies. In such distributed artificial intelligence societies each agent is an individual artificial intelligence capable of some useful inferences activity on its own, but being plunged in an artificial society communicating and co-operating with others it is able to enhance its performance [Bond and Gasser, 1988].

With the previous considerations on ontologies and knowledge-based systems, I already positioned my work in the symbolic branch of artificial intelligence. In the rest of this work, I shall only consider the symbolic branch of multi-agent systems as a paradigm to manipulate the symbolic knowledge I am interested in.

First, I shall introduce agents and multi-agent systems, the notions, the application fields and the special case of multi-agent information systems. In the second part, I shall survey methodological approaches and underline the importance of the organisational dimension in these approaches. Then, I shall discuss the needs for agent communication languages and the current proposals. Finally, I shall focus the survey on the multi-agent information systems, detailing some examples and abstracting some shared characteristics.

4.1 Agents and multi-agent systems

4.1.1 Notion of agent

Nearly every article on agent paradigm begins by explaining that programming progresses were achieved through higher abstraction enabling us to model systems more and more complex and that multi-agent systems have the potential to become a new stage in abstraction that can be used to understand, to model and to develop complex distributed systems. I agree, but this is one of the rare consensus of the domain. A consensus on a potential is not a sufficient foundation for a research field to be recognised and therefore, the definitional ground of multi-agent systems is the subject of numerous articles tackling the definition of the paradigm from numerous different points of views and backgrounds. However, I can safely say that agents are autonomous, loosely coupled software components exploiting different techniques of artificial intelligence. With references to [Ferber, 1999] I shall distinguish once for all the two main trends in the multi-agent systems domain:

- *distributed artificial intelligence*: started with distributed problem solving, usually based on symbol manipulation *i.e.,* influenced by the symbolic branch of artificial intelligence that follows a cognitivist paradigm and proceeds to create artificial intelligence based on the manipulation of symbols; these multi-agent systems are organisations of symbolic artificial intelligence components. This extreme is represented by the belief-desire-intention school ([Rao and Georgeff, 1991] and [Shoham,1993]) whose agents are called the *deliberative agents*. They comprise complex internal mental states models (their beliefs, their goals and their plans to achieve them) as well as sophisticated reasoning architecture, usually consisting of different modules that operate asynchronously [Sycara, 1998b], in order to derive the future actions and mental state from the current state of affairs. *Deliberative agents are usually used for strong processing architectures*.
- *artificial life*: simulation of life (usually swarming life). Follows a minimalist sub-cognitive approach of agents, where behaviours are based on stimulus-response, without complex internal representation, simple signal-based communication. They are called *reactive agents* and they have their roots in the work of [Brooks, 1991] and [Minsky, 1986]. The intelligence is created by an evolutionary process and based only on the emergent combination of simple interactions of agents with their environment. The main evolution criteria is viability usually applied by a simulation of a natural selection process. Their simplicity and high redundancy of roles make them robust and highly fault tolerant. *Reactive agents are usually used for simulation purposes*.

So I shall distinguish

- *heavy / coarse-grained agents* including complex internal systems (e.g. knowledge-based systems, logic-based planners, etc.) usually in small numbers in a system and having knowledge-level message passing communication.
- *light agents* with very simple internal structures (e.g. small automata) usually in large number to create an emergence of collective intelligence through stimulus and simple signal interactions.

Of course a multi-agent system can be anywhere on the continuum between these two extremes, and even combine different types of agents.

Ferber [1999] goes further, identifying the different trends in the domain: researches around the old distributed artificial intelligence school (cognitive MAS, small number of agents, coarse intelligent artificial autonomous systems, influenced by organisational sciences); researches on rational agents (following the agent internal architecture based on the distinction of Beliefs, Desires and Intentions); researches on speech acts and agent communication languages; distributed artificial intelligence and game theory and economics; Petri nets and multi-agent systems; parallelism and actor and multi-agent systems; reactive Agents; machine learning and multi-agent systems; etc.

Multi-agent systems are at the confluence of a variety of research disciplines and technologies, notably artificial intelligence, object-oriented programming, human-computer interfaces, and networking. [Sycara, 1998]. Their 'intelligence' and 'collective' dimensions led agents to be influenced by results stemming from extremely different disciplines, examples given by [Finin *et al.*, 1998] are:

- *Database* and *knowledge-base technology*; when agents need data and knowledge to represent and reason about things they have to do or manipulate.
- *Distributed computing*; concurrency is inherent to multi-agent systems, parallelism has to be engineered and interaction protocols have to be specified.
- *Cognitive science*; a BDI internal architecture modelling Beliefs about itself, other agents and its environment, Desires about future states and goals and Intentions about its own future actions.
- *Computational linguistics*; to build communication model, the most influent one being Speech Act theory.
- *Econometric models*; to predict and control the collective behaviour and equilibrium.
- *Biological analogies*; to reuse evolutionary process and natural selection for artificial life, genetic programming for adaptability, pheromones and environmental metaphor to build artificial ecologies.
- *Machine Learning*; applied to software agents to allow automatic adaptation to users and context.
- etc.

Needless to say that with such a diversity of contribution the agent and multi-agent system paradigm is being diluted in a multitude of perspectives. Thus, giving a definition is a perilous act in the actual field of Multi-Agent systems. If a consensus exists around the definition of the two core concepts of *agents* and *multi-agent systems*, it is one that says there is no consensus (yet). It seems that ontologists could find an excellent application field in the domain of agent-based system itself. I shall try here to review some propositions and set my own definitions for the rest of this work. I shall only consider the *heavy* or *deliberative* agents perspective.

[Singh and Huhns, 1999] identified some properties of agents including *autonomy*, *adaptability*, and *interactiveness*, but only to underline the fact that exceptions reveal that an agent need not have these properties. The same authors make it a compulsory condition for the agent to be capable of interacting with other agents at the social or communicative level. They distinguish social or communicative interactions from incidental interactions that agents might have as a consequence of existing and functioning in a shared environment. I shall also consider here that social interactions are a vital characteristic and that they require a communication ability for the agents.

A definition also calls for differential positioning with other existing notions, in particular objects and components; this first communication ability gives one of the strongest differences: "Like an 'object', an agent provides a message-based interface independent of its internal data structures and algorithms. The primary difference between the two approaches lies in the language of the interface. In general object-oriented programming, the meaning of a message can vary from one object to another. In agent-based software engineering, agents use a common language with an agent-independent semantics." [Genesereth and Ketchpel, 1994]. Agent-oriented interactions are conceptualised as taking place at the knowledge level [Newell, 1982].

Moreover, unlike objects, agents have control over their behaviour and decide if and how an action should be taken; agents embody a stronger notion of autonomy than objects. Once created, they are continually active while passive objects are quiescent most of the time, becoming active only when another object invoke one of their methods. The standard object-oriented programming model has nothing to do with the types of flexible autonomous behaviour analysed in agent-oriented programming: reactive, proactive, social [Wooldridge and Ciancarini, 2000].

In trying to answer the question: What is an Agent? [Wooldridge and Jennings, 1995] proposed to distinguish two notions of agent: a *weak notion* relatively uncontentious; and a *strong notion* potentially more contentious. The *weak notion* of agent denotes a computer system with the following properties:

- *autonomy*: it operates without the direct intervention of humans or others, and has some kind of control over its actions and internal state.
- *social ability*: it interacts with other agents (and possibly humans) via some kind of agent communication language.
- *reactivity*: it perceives its environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET, or perhaps all of these combined), and it may respond to changes that occur in it.
- *pro-activeness*: it does not simply act in response to its environment, it also is able to exhibit goal-directed behaviour by taking the initiative.

While this weak notion of Wooldridge and Jennings is a good candidate, in a multi-agent system the *pro-activeness* is not always exhibited by all the agents; it spreads into the whole system because some of the agents have this ability and trigger chain-reactions, or it is due to a user disguised by an interface agent, etc. In [Wooldridge, 1997] the pro-activeness is replaced by the ability to fulfil design objectives: "an agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives." Surprisingly the situatedness is not mentioned in the weak definition.

Concerning the strong notion of agency, in addition to having the above properties, the agent is either conceptualised or implemented using concepts that are more usually applied to humans. "For example, it is quite common in AI to characterise an agent using mentalistic notions, such as knowledge, belief, intention, and obligation. Some AI researchers have gone further, and considered emotional agents (...) Another way of giving agents human-like attributes is to represent them visually, perhaps by using a cartoon-like graphical icon or an animated face (...) - for obvious reasons, such agents are of particular importance to those interested in human-computer interfaces." [Wooldridge and Jennings, 1995]. We see that the strong notion of agency tends to be the intersection of all the aspects developed by the disciplines that influence multi-agent systems.

In [Jennings, 2000], agents are:

- *clearly identifiable* problem solving entities with well-defined boundaries and interfaces.
- *situated* in as environment they perceive through sensors and they act on through effectors.
- designed to *fulfil a specific purpose*; they have particular objectives to achieve.
- *autonomous*, they have control both over their internal state and over their own behaviour;
- capable of exhibiting *flexible problem solving behaviour* in pursuit of their design objectives; they need to be both reactive (able to respond in a timely fashion to changes that occur in their environment) and proactive (able to act in anticipation of future goals).

"Situated", is an interesting characteristic which, for me, is compulsory as soon as a paradigm has a distributed dimension. [Sycara, 1998] gives an interesting definition of an agent, underlying that it is a computer software system whose main characteristics are situatedness, autonomy, adaptivity, and sociability:

- *Situatedness* means that the agent receives some form of sensory input from its environment, and it performs some action that changes its environment in some way. The physical world and the internet are examples of environments in which an agent can be situated.
- *Autonomy* means that the agent can act without direct intervention by humans or other agents and that it has control over its own actions and internal state.
- *Adaptivity* means that an agent is capable of (1) reacting flexibly to changes in its environment; (2) taking goal-directed initiative, when appropriate; and (3) learning from its own experience, its environment, and interactions with others.
- *Sociability* means that an agent is capable of interacting in a peer-to-peer manner with other agents or humans.

There may be other attributes of agency given by [Wooldridge and Jennings, 1995], for instance:
- *mobility* is the ability of an agent to move around an electronic network; this definition is ambiguous with robot agent sitting in one computer system, but moving in the physical world.
- *veracity* is the assumption that an agent will not knowingly communicate false information.
- *benevolence* is the assumption that agents do not have conflicting goals, and that every agent will therefore always try to do what is asked of it.
- *rationality* is the assumption that an agent will act in order to achieve its goals, and will not act in such a way as to prevent its goals from being achieved at least insofar as its beliefs permit.

The characteristics used in the previous definitions are based on abilities which are finally tightly dependent on the actual role of the agents and by extension on the specifications of the system; thus they are very unstable. Yet they have the advantage of existing and of providing a starting point for discussion.

An interesting point of view, defended in a lecture by Yves Demazeau at the summer school EASSS 2000, is that there is no agent without a system, and that therefore, there should exist an external definition of an agent that does not rely on its individual characteristics. And indeed, the situatedness and social abilities must be respectively defined against the environment and social caracteristics of the multi-agent system. Another example of this view is found in [Genesereth and Ketchpel, 1994] where it is said that "an entity is a software agent if and only if it communicates correctly in an agent communication language.(...) More powerful agents utilise a larger portion of ACL; less powerful agents use a smaller subset. All are agents, so long as they use the language correctly."

I shall build my definitions from the ones given above, but if I detailed all these internal characteristics that can be attributed to agents, it is not so much for definition purposes, but rather for having an overview of the types of features that agent designers have been looking for so far. The agent definition debate is closely linked to the choice of the features to be considered as characteristic, distinctive, of an agent as opposed to a 'classical' piece of software. The list of characteristics that are debated gives an extremely interesting overview of the actual capabilities envisaged for agents. In Table 5, I give the most common abilities studied in the multi-agent system community so far. The diversity of abilities reveals the influence and the integration in multi-agent systems of results from different areas of research. It also explains that such a crossroad research field with so much overlaps with other areas has difficulties in finding its identity in consensual definitions.

This table calls for some remarks:
- Autonomy and Adaptability have been refined in sub-characteristics providing different degrees for these characteristics.
- Some of the abilities are closely related or depend on one another. For instance, rationality is defined in terms of goals and this means the agent is goal-oriented.
- Some of the abilities are conflicting (which in itself explains the difficulties to obtain a consensual definition based on abilities). For example 'benevolence' can be in opposition to 'self-control' where an agent can say no (especially if agent represent people with conflicting interests, involved in a negotiation).
- Some of them are at a very early stage of research (e.g. personality), others are well advanced or benefit from long-existing research field (e.g. reasoning).
- These abilities can be taken as building blocks to create higher levels of abilities. An agent architecture may combine abilities (e.g. reactivity and complex deliberation) in a concurrent fashion inside an agent and thus introduce a new hybrid ability. This type of considerations led some research groups to think that the agent architecture should be holistic.

Property	Sub-property	Meaning
Reactive		Responds in a timely fashion to changes in its environment.
Complex Mental State		The agent has knowledge, belief, intention, and obligation.
Graceful Degradation		Correctly handles a degraded operational state; it dies properly.
Temporally continuous (in the multi-agent system)		Once created, an agent is always running, but in a multi-agent system some agents may have short life while other are persistent.
Situated		The agent is located in an environment, perceives it through sensors and acts on it through effectors.
Autonomy		An agent takes initiative and exercises control over its own actions
	Individuality	Clearly identifiable artificial entity with well-defined boundaries and interfaces with its environment.
	Self-control	Does not blindly obey commands, but can modify requests, negotiate, ask for clarifications, or even refuse.
	Flexible	Actions are not scripted; the agent is able to dynamically choose which actions to invoke, and in what sequence.
	Goal-oriented (purposeful)	Accepts high-level requests indicating what another agent (human/software) wants and takes responsibility for deciding how and where to satisfy the requests on behalf of its requester.
	Proactive Self-starting	Ability to take the initiative; an agent can reason to decide when to act.
Personality		An agent has well-defined believable personality and emotional state that can improve interaction with human users.
Communication (socially able)		An agent can engage in complex communication with other agents (human/software), to obtain information or enlist help in accomplishing its goals.
Adaptability		An agent customises itself to the preference of its user and adapts to changes in its environment.
	Customisable Educable	Can be customised by another agent (most probably human) to meet preferences.
	Learning	Automatically customises itself and adapt to changes in its environment (user preferences, networks, information…) and modifies its behaviour based on its previous experience.
Migration		An agent can move from one system environment to another, possibly across different system architectures and platforms.
Mobility		An agent is embedded in a real world moving artefact.
Visual representation		The agent has a visual representation; this goes from classic interfaces to anthropomorphised and virtual reality ones.
Veracity		The agent will not knowingly communicate false information.
Benevolence		Agents do not have conflicting goals, and every agent will therefore, always try to do what is asked of it.
Rationality		An agent will act in order to achieve its goals, and will not act in such a way as to prevent its goals from being achieved - at least insofar as its beliefs permit.

Table 5 Agents characteristics adapted from [Etzioni and Weld, 1995], [Franklin and Graesser, 1996], [Nwana, 1996], [Wooldridge and Jennings, 1995], [Jennings, 2000] and [Sycara, 1998].

The choice of the abilities is linked to the needed roles identified for agents in the system: the stake is to combine the right abilities in the right behaviour to play a given role. The role analysis is the turning-point in designing agents as we shall see in the following section.

The characteristics with a greyed background are, in my opinion, the least common denominator of the agent definitions; my definition will be based on them.

Of course, the differences in agent definitions led to differences in agent internal architectures; depending on the characteristics of the agent, different internal agent architectures have been used. These architectures may be distributed over a continuum between purely reactive architectures and strong mentalistic architectures based on belief desires and intensions (and even lately emotions):

- The simplest internal architecture corresponds to the *purely reactive* characteristic of agents. It does not have any complex internal model of the environment, and is based using on a stimulus-response behaviour: it senses its environment and responds to the current state of affairs in a reflex fashion without any type of reasoning. The behaviour thus relies on two sequential functions: (1) *perceive*: that maps the external environment state to internal agent perceptions (2) *act*: that maps the internal perceptions to actions on the external environment. The purely reactive architectures are simple and quick, but they cannot be pro-active and they cannot learn from experience because they have no internal state. Thus as soon as a behaviour requires some memory or internal state abstraction of perceptions, pure reactivity has to be mitigated.
- The most complex internal architecture to date correspond to the *Belief, Desire and Intention* (BDI) architecture [Rao and Georgeff, 1991]. This is the rational agent architecture *par excellence*. This architecture adds complex logical internal representation and three high level functions to the reactive one (a) to manage the beliefs an agent maintains about its environment, (b) to form desires that motivate the agent behaviour and (c) to build plans to achieve these desires. The complete architecture consists of: (1) *perceive*: that maps the external environment state to internal agent perceptions (2) *believe*: that builds new beliefs from perceptions and current beliefs. (3) *desire*: derives desires from the current set of beliefs and intentions (4) *intend*: build intentions motivated by the current beliefs and desires (5) *act*: that maps the internal intentions to actions on the external environment. Of course this powerful architecture has for cost the difficulty of designing and implementing a complete BDI behaviour.

Not surprisingly a lot of intermediate architectures try to propose intermediate options providing greater internal capabilities than reactive architectures and less complexity than BDI ones. As usual the different options for agent architectures led to hybrid solutions. Among the existing alternative or hybrid options, I shall mention:

- *Finite State Machine* and Petri nets: the behaviours of an agent are defined by an abstract machine consisting of a set of states (including the initial state and some terminal states), a set of possible input events that correspond to perceptions, a set of possible output events that correspond to actions, and a state transition function. The function takes the current state and an input event and returns the new set of output events and the next state; the whole can be represented in a state transition diagram. Another way of representing state-based behaviours are Petri nets. In any case, this architecture introduces an additional step in the reactive one, usually called *next*, consisting of an intermediary decision step between *perceive* and *act*, where the finite state machine decides the following state from the perceptions which may result in triggering some actions. Since an agent can be involved in several social roles at the same time, a Concurrent Finite State Machine architecture can be an easy way to implement concurrent agent interactions. The main drawbacks are the lack of goal-oriented, proactive behaviour and the fixed structure of the finite state machine that is most of the time hardwired in the agent source code. Additional architecture features are needed to handle interactions between behaviours if they may occur.
- *Connectionist* and *Machine learning*: neuronal networks and machine learning techniques may be used to train the behaviour of an agent; instead of designing a behaviour, a learning behaviour is created and trained over examples to obtain the needed ability: recognising, classifying, etc.

- *Knowledge-based*: intermediary step before BDI, these architectures use knowledge-based system architecture inherited from expert-systems. Agents have an individual knowledge base fed by analysing external perceptions, an inference engine and an individual base of inference rules that determine their behaviour. This architecture is descended from classical long established reasoning systems [Haton *et al.*, 1991]. Moreover, this kind of agent architecture allows to simply implement primitive, but efficient autonomous and pro-active behaviours. Rules and knowledge enable both reactions and more proactive behaviours; production rules are intentions and knowledge constitute beliefs, the desire may be explicit (BDI) or implicit (in the inference design).

Hybrid architectures are divided into two main categories:
- *Parallel layers architecture*: the software layers corresponding to different behaviours are each directly connected to the sensory input and action output; the problem is the management of the competition for some resources or to perform actions. In this case, there are two possible approaches: (1) layers subsumption based on priority links between the layers to impose the result of a layer over another one in case of conflict (2) meta-control: an independent overseeing module resolves the conflicts or handles the priorities. Blackboard architecture is a kind of meta-control where independent components are not directly connected, but share a repository of data (the blackboard) and a meta-controller handles conflict of access to the blackboard [Lesser and Corkill, 1983] [Haton *et al.*, 1991]. Competitive tasks are another example where one task at a time is activated and chosen by a decision mechanism analysing the current state [Drogoul and Ferber, 1992].
- *Hierarchical layers architectures*: the software layers corresponding to different behaviours are stacked up. Two options can be identified: one pass control and two pass control architectures. In one-pass architectures, the control flows sequentially through each layer until the final layer generates action output [Ferguson, 1995]. In two pass control architectures, information flows up the architecture and then control flows back down.

4.1.2 Notion of multi-agent systems

A system with one agent is classical artificial intelligence; a system with multiple agents is an artificial society. Thus, the main issues and the foundations of distributed artificial intelligence are the organisation, co-ordination and co-operation [Gasser and Huhns, 1989]

Extending the definition of distributed problem solvers of [Durfee and Lesser 1989], a Multi-Agent System is defined as a loosely coupled network of agents that work together as a society aiming at solving problems that would generally be beyond the reach of any individual agent. This definition is, *mutatis mutandis*, the general definition that is usually given in the major part of articles and inherited from the field of distributed planning.

According to [Sycara, 1998b], the characteristics of a multi-agent systems are that:
- each agent has *incomplete information or capabilities* for solving the overall problem tackled by the system and, thus, has a limited viewpoint.
- there is *no system global control*: the collective behaviour is the result of social rules and interactions and not of a supervising central authority.
- *resources are decentralised*: resources needed for the completion of the tasks assigned to the system are divided and distributed.

[Zambonelli *et al.*, 2000] distinguish between two main classes of multiple agents system:
- *distributed problem solving systems* in which the agents are explicitly designed to co-operatively achieve a given goal in a benevolent fashion.
- *open systems* in which agents are not necessarily co-designed to share a common goal and can dynamically leave and enter the system.

In the former case, all agents are known *a priori* and benevolent to each other and therefore, they can trust one another during interactions. In the latter case, the dynamic arrival of unknown agents needs to be taken into account, as well as the possibility of self-interested behaviour in the course of the

interactions; additional agent description capabilities are vital. However, in any case, multi-agent systems research is concerned with the problem of co-ordinating the behaviour among a collection of intelligent agents *i.e.,* what are the mechanisms to co-ordinate the agent knowledge, goals, skills, and plans jointly to take action and to solve problems [Bond and Gasser, 1988]. Thus, the multi-agent dimension brings additional architectural problems to traditional artificial intelligence:
- *organisational structure* of the agent society and its patterns (hierarchy, anarchy, etc.).
- *organisational interactions* sustaining the structural patterns.
- *organisational and environmental rules* constraining the structures and the interactions.

In a multi-agent system, several agent types implementing different internal agent architectures can collaborate; this is the reason why multi-agent system are intrinsically prone to integration. This also corroborates the difficulty of agreeing on an agent definition, since the major part of multi-agent systems are heterogeneous in the compliance of their agents to a fixed agency notion. For instance, sophisticated planning agents and barely reactive agents can live together in the ecology of a multi-agent information system, providing services from mere wrapping to complex mining, etc.

Once again, we can distinguish:
- *emergent approaches* where the structure is not directly engineered, but influential mechanisms are designed to ensure a convergence of collective structuring and interactions towards a configuration that meets the application objectives (simulation, global task, etc.).
- *fixed design* approaches where the structure and interactions are fixed to obtain a wanted architecture that fulfils application specifications.

By introducing fixed point in emergent approaches or adaptive characteristics in fixed design, a whole spectrum of architectural design approaches have been proposed; I shall detail this in the section on multi-agent design apporaches.

4.1.3 Application fields and interest

The range of application domains of agents is impressive and probably caused the hype that brought so many criticisms upon the field. Examples of applications are:
- *E-commerce* assistance: agents are used for mediating negotiations, comparing proposals, etc.
- *Simulation* of real distributed systems: agents as a modelling tool (biology, traffic regulation etc.)
- Portable and *mobile devices*: agents manage mobile systems (cellular phones, PDA, etc.) to deal with connectivity problems, delegation of information search tasks, etc.
- *Robots* and robot co-operation: agents are embodied in robots and interact in the real world (space missions, hazardous interventions, etc.)
- *Interfacing* people: Interface agents (facing the user), personal assistants (user profiling, learning, etc.), believable agents that communicate and interact with humans (virtual reality, graphics, avatars, etc.)
- *Integration and interoperation*: agents wrapping existing legacy systems (enterprise integration, information systems, etc.)
- *Critical application systems*: agents redundancy and concurrency is used to ensure efficiency, reliability, flexibility and robustness.
- *Design and maintenance*: introduction of new agents can ensure extensibility, maintainability in applications without even turning off the system; it also provides separation of concerns.
- Elder independence, health care, manufacturing and supply management, etc.

Of course, I shall make a special case for one application domain which the one of this work *i.e.,* information, knowledge and expertise distribution management. It can be divided in two aspects:
- *distributed problem solving* involving real distributed actors (e.g. scheduling, air-traffic control, expert matching, manufacturing to process control, collaborative work),
- *management of distributed resources and means* (e.g. network balancing, sensors data, information gathering)

It is in this area that *information management agents* are situated.

4.1.4 Notion of information agents and multi-agent information systems

Leaving aside the general agent definitional problems, a less contentious matter is the definition of agents and multi-agent systems dedicated to information. Multi-Agent Systems paradigm, when applied to information systems, meets the problems addressed in Collaborative Information Systems and leads to a specialised kind of multi-agent systems: Multi-agent Information Systems (MAIS). In this section I shall first identify the original issues and research fields that started the multi-agent information field. Then, I shall try to give a general overview of the present characteristics of the field, its agents and its systems. Finally, I shall make the link with organisational information systems.

4.1.4.1 Tackling the heterogeneity of the form and the means to access information

In the mid 90s, ontologies were beginning to show their importance for information systems. For instance, in [Mena *et al.*, 1996], the authors acknowledge the critical need to complement traditional browsing, navigation and information retrieval techniques with semantics-based strategies. The growing importance of ontologies in information system will find its symmetry in the growing importance of ontologies in multi-agent systems, especially for the problem of semantic message passing between deliberative agents.

At that time, [Mena *et al.*, 1996] proposed to divide the various approaches of information systems in three types:
- The *classical keyword-based* and navigational approaches with no semantics, but simple to use and to support and highly scalable.
- Approaches based on a *global shared ontology* which supports expression of complex queries and indexes. It is conceivable when the fact of having one and only one ontology is conceivable. If several ontologies already exist and must be modified or integrated, the following option may be better.
- Loosely coupled approaches where interoperation across *several ontologies* is achieved via mappings between them. The system is more complex and answers tend to be inferior to the previous one, but they are much more scalable and extensible.

The multi-agent systems inherited from these various approaches. Many requirements detected at that time and components developed to meet them found their counterpart in multi-agent information systems.

The trend was not only to homogenise access to heterogeneous information repositories, but to homogenise access to resources and services in general. One of the famous first projects which started to use the term of Agent was the Softbot. The Softbot [Etzioni and Weld, 1994] was trying to unify access to available services, integrating the basic services in a user-friendly agent capable of planning and relieving the user from some tedious tasks and sometime obvious enough for a planner to solve them without calling on the user if not necessary. The softbot used a UNIX shell and the World-Wide Web to interact with a wide range of services including ftp, telnet, mail, file systems, archie, gopher, netfind. The idea of unifying the access led to the encapsulation of existing resources in an agent. The trend to 'agentise' existing software is now a very active research activity too, IMPACT is an example of the projects interested in this area [Arisha *et al.*, 1999].

Individual situatedness and locality of agents in general enable information agents to adapt to local characteristics of the information resources and means they have at their disposal.

4.1.4.2 Tackling growth and distribution of information

The diversity of software (heterogeneity of programs, languages, interfaces, data format) lead to a demand for interoperation between these tools "to exchange information and services with other programs and thereby solve problems that cannot be solved alone" [Genesereth and Ketchpel, 1994] Software agents mitigate an information environment's heterogeneity by interacting through common protocols, and manage its large size by making intelligent local decisions without centralised control [Huhns, 1998].

The growing complexity and size of information resources available online is fast outstripping the manual browsing and searching centralised solutions. A long-term approach to this problem is to design intelligent information agents that provide active assistance in the process of finding and organising information [Lieberman, 1999]. These agents differ from conventional information retrieval solutions by the fact they interact with the user and other agents to defeat the dynamic nature of the information they deal with. Multi-agent information systems oppose their heterogeneity and dynamic nature to the one of the fast growing information landscapes. The different tasks of information management are delegated to different software agents that act on behalf of the user.

The organisational dimension of multi-agent systems in general enables multi-agent information systems to collectively capitalise huge mass of distributed information resources through communication and co-operation.

4.1.4.3 Transversal issues and characteristics

[Lieberman, 1999] termed this category of software agents: 'intelligent information agents', but specifies that by *intelligent*, we do not mean that the agent is as intelligent as a human. However, there are many kinds of intelligent behaviour for which it is feasible to design artificial simulations and Lieberman gives the examples of simple learning, inference, user modelling and context sensitivity, etc.; the key problem being to understand when and how limited forms of intelligence can be of real assistance.

The shift in the software design approach is to go beyond the tool and object paradigm to a tool/object/agent paradigm where systems do not only provide software tools and objects to be manipulated by users, but also agents assisting the users by automating these manipulations. Computers enabled us to tackle some aspects of the complexity of our real world, but doing so, they created a virtual world the complexity of which is growing and become in its turn hardly manageable by human intervention alone. So it is time to populate these worlds with intelligent software agents in charge of managing it.

The stake is to obtain a society of trusted and reliable agents that will act on our behalf in the information landscape. [Klusch, 1999d] gives examples of tasks we could envisage to delegate to them:

- search for, acquire, analyse, integrate, and archive information from multiple heterogeneous distributed sources, inform users when new data of special interest become available,
- negotiate for, purchase and receive information, goods and services,
- explain the relevance, quality and reliability of that information, and
- adapt and evolve to changing conditions.

The multi-agent system approach for information systems is not a wishful thinking. It is based on a number of considerations that justify the interest of having distributed intelligent systems. Such considerations can be expressed as follows:
- Information resources available on-line are inherently distributed.
- Supporting architecture need to share capabilities and resources.
- Solutions must acknowledge and hide the underlying distributed problem solving complexity.
- Heterogeneity and complexity call for separations of concerns an modularity for design and maintenance.
- Integration of heterogeneous services and flexibility call for local adaptability and global co-operation.
- No central control should be used to avoid bottleneck and central weak spot; the architecture should ensure a maximum of the services in degraded conditions.

Until recently, in the literature, the definition of an information agent was usually limited to agents for information retrieval *i.e.,* agents that will fetch information for a user or a group of users. For instance [Klusch, 1999a] also emphasised this aspect: "Roughly speaking, information agents are computational software systems that have access to multiple, heterogeneous and geographically distributed information sources. Such agents may assist their users in finding useful, relevant information; in other words, managing and overcoming the difficulties associated with 'information overload'. Information agents not only have to provide transparent access to many different information sources in the Internet, but also to be able to retrieve, analyse, manipulate, and integrate heterogeneous data and information on demand, preferably in a just-in-time fashion".

[Singh and Huhns, 1999] rightly extend this notion stating that information agents do not only help find information: their broad role is to manage information. This view is not inconsistent with current work, but it is important to be aware of this distinction and not to limit the range of possible roles. More recently [Klusch, 2001] provided an extensive survey on information agents where can be found the following definition: an information agent is an autonomous, an intelligent agent that has access to one or multiple, heterogeneous and geographically distributed information sources, and which pro-actively *acquires*, *mediates*, and *maintains* relevant information on behalf of users or other agents preferably just-in-time. However, as I already stressed it for the general agent definition, the pro-active characteristic may not be exhibited by all the agents of a system.

Thus following [Klusch, 2001], I shall say that to be an information agent, an agent is supposed to satisfy one or several of the following requirements:
- *Information acquisition*: provide access to information sources eventually wrapping or translating them and proposing retrieval, extraction, analysis, filtering or monitoring services.
- *Information management*: update and maintain the content of an information source.
- *Information search*: find and contract the information services needed to answer a request.
- *Information integration*: provide services for merging heterogeneous information.
- *Information presentation*: format and present information in an adequate way.
- *Information adaptation*: adapt information services and results to the context and the users

These requirements led Klusch to distinguish possible needed skills for information agents. The author starts by differentiating between communication, knowledge, collaboration, and rather low-level task skills. Klusch depicted it as shown in Figure 31, with some key enabling technologies listed below each of the different types of skills.

Communication skills of an information agent comprehend:
- *communication with information systems*: this requires APIs e.g. JDBC for databases,
- *communication with human users*: it is either done through multimedia channels (computer screen, speech generation and recognition or other signal processing, etc.),
- *communication with other agents*: this requires the use of an agent communication language (ACL) and a middleware platform ensuring communication services.

Knowledge skills rely on different results of traditional artificial intelligence: knowledge-based systems and knowledge engineering, metadata and user profiling, natural language processing tools, machine learning techniques, etc.

Collaboration is divided into:
- *Collaboration between information agents*: collaboration at the knowledge-level that may rely on middle agent services and established protocols.
- *Collaboration with human users*: based on techniques stemming from human-computer interaction and affective computing.

Task level skills are the core information processing activities of multi-agent information systems. Agents performing these tasks use results from: information retrieval and filtering, information extraction and data mining (text mining, web mining, Bayesan techniques etc.), information integration (ontology-based and database systems), etc.

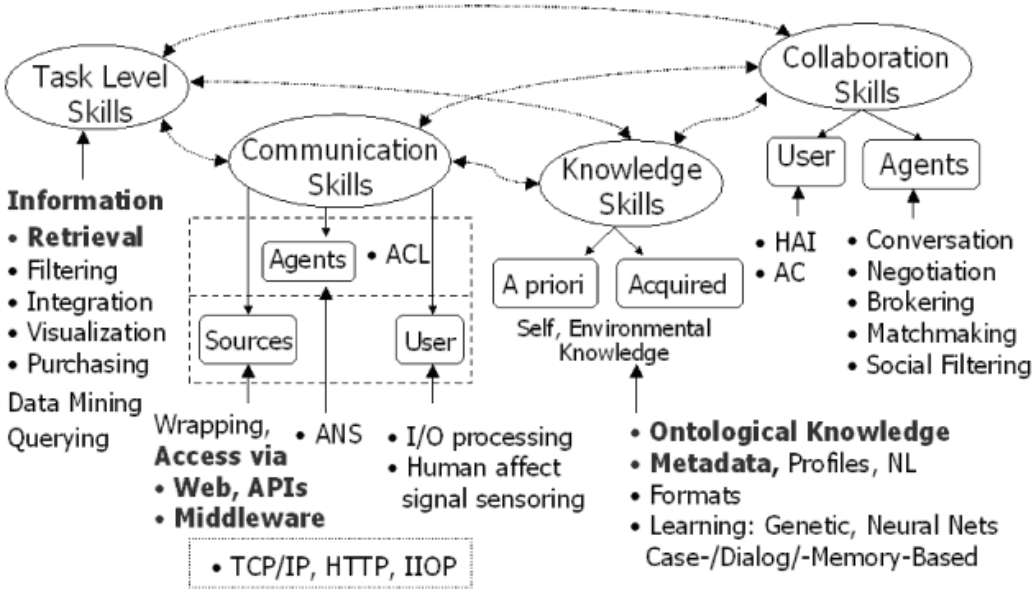


Figure 31 Information agents basic skills [Klusch, 2001]

[Klusch, 2001] also distinguishes between three prevalent approaches for building information agents:
- *User Programming*: information agents are programmed by their user from scratch using rules for processing information. While this approach offers great flexibility, it requires too much insight, understanding and effort from the user. Hybrid solutions can deliver prototypical agents, or highly customisable agents.
- *Knowledge engineering*: information agents are endowed *a priori* with application-specific knowledge about. It requires substantial efforts from knowledge engineer and the agent is highly specific to the application; however knowledge-based approaches are starting to provide supporting tools to ease construction, reuse and maintenance of knowledge bases and the Semantic Web is an effort in trying to build largely available ontologies and annotations to guide agents.
- *Machine learning*: information agents automatically acquire the knowledge they need to assist the user. These methods usually require large samples to be able to abstract and learn useful patterns in the usage of the system. Combining several techniques for different aspects seems to be interesting (on the fly learning of preferences, collective off line clustering of profiles, etc.)

4.1.4.4 The case of organisational information systems

The field of multi-agent information systems is very active and most promising application areas are, among others, distributed Web-based collaborative work, information discovery in heterogeneous information sources *and intelligent data management in the Internet or corporate intranets* [Klusch, 1999a].

[Papazoglou and Heuvel, 1999] analysed in details the relation between the business and its information system. Authors showed that the modifications occurring in corporate structures as we saw in the first chapter also call for the deployment of a more sophisticated middleware infrastructure that supports company-wide and cross company distributed information processing activities. "This infrastructure should naturally promote interoperation between web-enabled applications and a large number of interconnected legacy information systems, and satisfy a request, irrespectively whether the business logic lies in the legacy system or in an Internet-enabled workflow transaction" [Papazoglou and Heuvel, 1999]. The ideas appearing here suggest the mapping of human organisational structures and activities to organisational structures and activities of corporate information systems.

Moreover, in parallel to information overload there has been and there is still an increasing industrial interest in the capitalisation of corporate knowledge, leading to the development and deployment of knowledge management techniques in more and more companies. As explained before, the coherent integration of this dispersed knowledge in a corporation is called a corporate memory. Most of the initiatives in the domain were intranets and corporate webs. Thus corporate memories are facing the same problem of relevance as Web search engines when retrieving documents because the information landscape of a company is also a distributed and heterogeneous set of resources. Therefore, following the same considerations, it seems interesting to envisage a distributed and heterogeneous system such as multi-agent information system to explore and exploit the information landscape of corporate intranet and extranets. The purpose is to allow the information sources to remain localised and heterogeneous in terms of storage and maintenance, while enabling the company to capitalise an integrated and global view of its corporate memory. The multi-agent system approach allows users to be assisted by software agents usually distributed over the network. These agents have different skills and roles trying to support or automate some tasks: they may be dedicated to interfacing the user with the system, managing communities, processing or archiving data, wrapping legacy systems of the company, etc.

Authors of [Papazoglou and Heuvel, 1999] believe that co-operative information systems have an unprecedented impact on modern organisations and will have particular influence on how co-operative business will be conducted in distributed organisations. As explained, enterprise modelling will enable the system to get insight in its environment, but the other point I am making here is that the enterprise configuration will also guide the architecture, the configuration and the rollout of the information system: it will become a strong link between the corporate information system and its environment, shaping the system and driving its interactions, having string impact on how the system effectively assist the organisational activities.

[Papazoglou and Heuvel, 1999] recognise the co-operative systems will help tackle distribution of data, distribution of control, distribution of expertise and spatial distribution of resources in an organisation. Co-operative information systems are "highly dynamic environments. They address a set of problems that are beyond the capabilities and expertise of individual component information systems and which require the synergy of a collection of component information systems to be solved. In these environments, information agents should know about their capabilities and should exchange knowledge and information with other such systems to achieve co-operation." [Papazoglou and Heuvel, 1999]

4.1.5 Chosen definitions

Based on these elements of the state of the art and my understanding of the distributed artificial intelligence literature, I shall use the following definitions in my Ph.D.:

agent	An agent is a clearly identifiable **individual artificial entity** with well-defined boundaries and interfaces. It is **situated** in an environment; it perceives it through sensors; it acts on it and **reacts** to it through effectors. It has **social abilities** to interact in with other agents or humans. Meanwhile it keeps its **self-control** allowing it to modify requests, negotiate, ask for clarifications, or even refuse.
information agent	An information agent, is an agent providing at least one of the following services: information acquisition (to provide access to information sources and eventually some value-added services), information management (to update and maintain the content of an information source), information search (to find and contract the information services needed to answer a request), information integration (to merge heterogeneous information), information presentation (to format and present information in an adequate way) information adaptation (to adapt information services and results to the context and the users).
multi-agent system (MAS)	A multi-agent system is a loosely coupled network of agents that work together as a society aiming at solving problems that would generally be beyond the reach of any individual agent.
heterogeneous multi-agent system	An heterogeneous agent system refers to an integrated set-up of at least two or more agents which belong to two or more different agent classes. [Nwana, 1996]
multi-agent information system (MAIS)	A Multi-Agent Information System (MAIS) is defined as multi-agent system aiming at providing some or a full range of functionality for managing and exploiting a distributed information landscape (see information agent).
open system	An open system is one in which the structure of the system itself is capable of dynamically changing [Hewitt, 1986].
open multi-agent system	An open multi-agent system is a multi-agent system in which agents types are not necessarily co-designed and agent instances dynamically leave and enter the system at run-time.
closed multi-agent system	A closed multi-agent system is a multi-agent system in which the agent types are explicitly known and usually co-designed before starting the system; new agents can be added or removed at run-time by administrators and in a controlled fashion.
corporate multi-agent information system	Usually a closed multi-agent information system supporting the information management and exploitation tasks linked to the activity of an organisation. Gateways with external and/or open systems may be engineered to manage interactions with external sources.

Table 6 Definitions used in multi-agent system engineering

4.2 Design rationale for multi-agent systems

Multi-agent systems offer a new paradigm for building complex software thanks to the organisational abstraction they provide. The other side of the coin is that we need a methodology to design these complex systems as organisations of co-working agents. I shall review here a number of organisational design approaches where the multi-agent system architecture is tackled, as in a human society, in terms of groups, roles and relationships. The manifesto of [Panzarasa and Jennings, 2001] goes even further, advocating the application of modern organisation theory. Of course this type of approach is even more attractive in the context of an artificial society designed to assist a human society.

4.2.1 Overview of some methodologies or approaches

"The role of agent-oriented methodologies is to assist in all the phases of the lifecycle of an agent-based application, including its management." [Iglesias *et al.*, 1999]. I started this section studying the surveys of [Iglesias *et al.*, 1999] and [Wooldridge and Ciancarini, 2000] which mainly concentrated on methodologies that extend existing object-oriented and knowledge engineering methodologies, which represent the major part of the actual methodologies. I shall not give as many details as the other surveys do. I shall just concentrate on the aspects relevant to forthcoming discussions and justifications of my work.

4.2.1.1 AALAADIN and the A.G.R. model

[Ferber and Gutknecht, 1998] proposed AALAADIN and the A.G.R. (agent/group/role) model depicted in Figure 32. In this model an agent is only specified as an active communicating entity which plays roles while organisational concepts such as groups, roles, structures and dependencies, are first-class citizen of a design methodology.

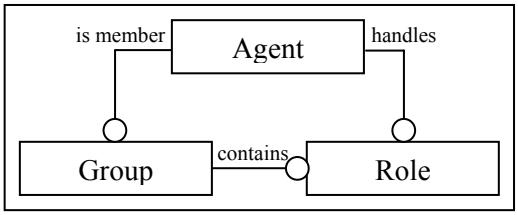


Figure 32 The AGR model [Ferber and Gutknecht, 1998]

This is one of the approaches where the organisational metaphor is the most exploited: agents can join and leave as many organisations as they need, playing different roles in the different groups.

The model is divided into:
- *concrete level concepts*: agent, role, group
- *abstract level concepts*: possible roles, possible interactions, possible organisational structures.

Groups are defined as atomic sets of agent aggregation that can freely overlap. Roles are abstract representations of an agent function, service or identification within a group; an agent can handle multiple roles. An organisation is viewed as a framework for activity and interaction through the definition of groups, roles and their relationships. The organisation is described solely on the basis of its structure, *i.e.,* by the way groups and roles are arranged to form a whole, without being concerned with the way agents actually behave. Organisation modelling aims at abstracting away from the interactions between the agents of a complex multi-agent systems and their fundamental and recurrent patterns.

Different types of requirement patterns are defined and formalised (single role behaviour requirements, intergroup role interaction requirements, intragroup communication successfulness requirements, and intragroup role interaction requirements) as well as the logical relationships between these different types. Formalisation and proofs rely on π-calculus.

4.2.1.2 AOR

A.O.R. stands for Agent-Object-Relationship [Wagner, 2000] and stems from the domain of relational databases. It is an agent-oriented approach for the analysis and design of organisational information systems. In A.O.R. modelling, an entity is either an object, an agent, an event, an action, a claim, or a commitment. Special relationships between agent classes and event, action, claim and commitment types supplement the fundamental association, composition and specialisation relationship types of Entity-Relationship modelling.

The emphasis that really interests me in this method is the ontological distinction between agents and objects, the existence of instances of both types and their relations. Only agents can communicate, perceive, act, make commitments and satisfy claims. Objects do not communicate, cannot perceive anything, are unable to act, and do not have any commitments or claims. Types of agents and objects are organised in subsumptions and partonomies.

An organisation is viewed as a complex institutional agent managing the rights and duties of its subagents that act on behalf of it, and being involved in a number of interactions with external agents; this aspect of the approach is quite close to holonic perspective. An organisational information system, then, is viewed as an artificial agent possessing a global view of the organisation and interacting both with subagents and with external agents.

The author proposes an algorithmic transformation of an A.O.R. model into a database schema.

4.2.1.3 ARC

ARC stands for Agent Role Component [Fan, 2000]. It is a bottom-up compositional approach inspired by component-based approach: an agent is composed of roles, a role is composed of components. The method aims at facilitating reuse proposing component-based roles for agents.

The agent controls the roles and provides the beliefs, desires and intentions. The role assembles the component based on task hierarchy diagrams. The method is highly formalised and uses associated diagrams to support the design:
- task hierarchy diagrams
- plan graph
- extended sequence diagram
- use cases
- etc.

4.2.1.4 ARCHON

ARCHON stands for ARchitecture for Cooperative Heterogeneous ON-line systems [Varga *et al.*, 1994] [Jennings *et al.*, 1996]. It was a large European project on distributed artificial intelligence. Its analysis combined:
- a *top-down approach*, that identifies the system goals, the main tasks and their decomposition,
- a *bottom-up approach*, that allows the reuse of pre-existing systems, constraining the top-down approach.

As in many approaches, there are two design levels; the agent community design and the agent design. Interestingly, the design steps are:
1. first the design of the agent community to define the agent granularity and the role of each agent
2. then, the design of the user interfaces.
3. finally, the skills and interchanged messages are listed and the agent design encodes the skills for each agent.

It is interesting to see the intermediate step with interfaces denoting that both agent-agent interactions and human-agent interactions must have been taken into account before considering individual agent modelling.

4.2.1.5 AUML

AUML stands for Agent UML [Bauer *et al.*, 2000] and extends the well-known object-oriented notation for agent design. Just like its object-oriented counter-part, AUML is not a methodology, but a notation for design documentation.

[Odell *et al.*, 2000] discussed several ways in which the UML notation might usefully be extended to enable the modelling of agent systems. The extensions they proposed include:
- support for expressing concurrent threads of interaction (e.g. broadcast messages) to enable designers to model such well-known agent protocols as the Contract Net.
- a notion of *role* that extends that provided in UML, and in particular, allows the modelling of an agent playing several roles.

UML is suitable to define traditional systems, but because of the proactive behaviour of some agents, the behaviour of the system itself is not completely driven by user actions. For this reason designers have to use dynamic diagrams (Use Case, Sequence, Collaboration and State Diagrams) with agents as external actors when they show a pro-active behaviour.

[Bergenti and Poggi, 2000] propose UML-based notation to represent the models of:
- the *architecture* of the multi-agent system,
- the *configuration* of a multi-agent system (through deployment diagrams),
- the *ontology* followed by the agents,
- the *protocols* used to co-ordinate the agents (interaction diagrams),
- the *roles* played by the agents.

It can be supported by any off-the-shell CASE tool. In my opinion, the ontology and role diagrams may be too shallow to represent such complex objects, however other diagrams have be successfully used.

4.2.1.6 AWIC

The AWIC stands for Agent/World/Interoperability/Co-ordination [Müller, 1996]. It is an hybrid design method based on several software engineering ideas including strategies of simulation systems building. It was used in domains such as transport management, traffic control, distributed databases, and co-operating expert systems.

It is an iterative design approach based on five models:

A	*model*: Identification of the agents in the problem domain. Specification of their tasks, perception, activities, world knowledge, and basic planning abilities.
W	*model*: Representation of the world (environment) the agents will act in. The activities defined in the A-model have to be offered as services and general laws have to be provided which prevent the agents from acting harmfully.
I	*model*: Definition of the interoperability between the world and the agents. *i.e.,* the action requests (from agent to world) and perceptual information (from world to agent).
C	*model*: Specification of the agent co-ordination through inter-agent communication, extension of the planning abilities towards joint planning and joint activities.

The general idea is to fill these models in a sequence of iterations. Each iteration is followed by a cross-checking phase where the models are analysed concerning consistency and over-specification.

A task oriented design method is used to define the A-model and to check the consistency of the interaction (co-ordination) activities in the C-model. This method comprises four steps:

1. The identification of the tasks that agents must perform (this step copes with the problem of defining the interface between the agents and the external world, *i.e.*, the hardware environment, the user interfaces, etc.);
2. The decomposition of complex and distributed tasks in tasks that can be managed by single agents;
3. The assignment of the tasks to the appropriate agents;
4. The definition of the interactions among the agents to perform complex and distributed tasks and to avoid conflicts between different tasks.

UML-Use Case diagrams may be used in the I-model and the C-model in order to describe the interactions between the respective entities.

4.2.1.7 Burmeister's Agent-Oriented Analysis and Design

The agent-oriented approach proposed in [Burmeister, 1996] is based on OMT (Object Modelling Technique) [Rumbaugh *et al.*,1991] and CRC cards (Class-Responsibility-Collaborator cards which characterise objects by class name, responsibilities, and collaborators) used to document collaborative design decisions.

This methodology defines three models for analysing an agent system and the process steps for the development of each model:

- *Agent model*: it identifies the agents and detail their internal structure; agents and their environment are documented using an extension of CRC cards for including beliefs, motivations, plans and co-operation attributes.
- *Organisational model*: it describes the relationships between agents (inheritance and roles in the organisation); roles of each agent are analysed and documented using OMT notation for the inheritance hierarchy and the relationships between the agents.

- *Co-operation model*: it describes the interactions between agents; co-operations and involved partners are identified, and the types of interchanged messages and used protocols are analysed and documented.

4.2.1.8 CoMoMAS

The CoMoMAS stands for Conceptual Modelling of Multi-Agent Systems [Glaser, 1996]. It started as an extension of the knowledge modelling methodology CommonKADS [Breuker and Van de Velde, 1994] in order to apply it to the design of multi-agent systems. It now proposes a complete methodology and the latest version [Glaser, 2002] includes an engineering environment for the development of multi-agent systems.

Since it was influenced by a knowledge engineering perspective, agents are seen as interacting entities having different kinds of knowledge; the methodology tries to identify this knowledge for each agent during the design.

CoMoMAS proposes a complete set of conceptual models and the associated methodology to guide the overall development process from design to validation:
- *Agent model*: this is the core model of the methodology. It defines the agent architecture and the knowledge they manipulate. The knowledge is classified as social, co-operative, control, cognitive and reactive knowledge.
- *Expertise model*: describes the cognitive and reactive competencies of the agent. It distinguishes between task decomposition knowledge (for agents and for users), problem solving knowledge (methods and the strategies to select them) and reactive knowledge (stimuli/response).
- *Co-operation model*: describes the co-operation between the agents using conflict resolution methods and co-operation knowledge (communication primitives, protocols and interaction terminology).
- *System model*: defines the organisational aspects of the agent society together with the architectural aspects of the agents.
- *Design Model*: collects the previous models in order to operationalise them, together with the non-functional requirements.

4.2.1.9 Cassiopeia methodology

Cassiopeia [Collinot *et al.*, 1996] is an organisation-centric analysis method for designing a multi-agent system. It was used as a methodological framework for designing and implementing the organisation of a soccer robot team, studying collective phenomena of organisation in robot societies.

It is a role-based method that intends to meet three points:
- independence from the implementation techniques;
- definition of an agent as a set of three different levels of roles;
- specification of a methodological process that reconciles both the bottom-up and the top-down approaches to the problem of organisation.

However, the method is essentially bottom-up in nature and consists of three steps:
- *identify the elementary behaviours* that are implied by the overall system task. Behaviours are listed using functional or object oriented techniques.
- *identify the relationships* between elementary behaviours. Relational behaviours are analysed and dependencies between the agents are studied using a coupling graph.
- *identify the organisational behaviours* of the system, in particular the way in which agents form themselves into groups. The dynamics of the organisation are described and analysed using the coupling graph, especially which agent can start or end a coalition and how it does.

Authors have acknowledged that a large part of the design activity is nevertheless left to heuristic choices or experimental work; this remark is of course not limited to Cassiopeia.

4.2.1.10 DESIRE

DESIRE [Brazier *et al.*, 1997] [Dunin-Keplicz and Treur, 1995] provides a method to study and design compositional multi-agent systems for complex and distributed tasks. It is a component-based top-down decomposition approach oriented by task compositional structure. It proposes a description language for intra-agent and inter-agent structure. The modelling framework consists of:
- a design method for compositional multi-agent systems,
- a formal specification language for system design,
- a graphical notation for specifying such compositional systems and the associated graphical editor
- software tools to support system design,
- an implementation generator to automatically translate specifications into code in a specific implementation environment,
- verification tools for static properties of components such as consistency, correctness, completeness.

Specifications are based on the notion of compositional architecture: an architecture composed of components with hierarchical relations between them. Each of the components has its own input and output interface specification and task control knowledge, specifying the dynamics of the whole system in a structured, decentralised manner. Components (composed or primitive) can be (re)used as building blocks. A complex task is modelled as a composition of a number of tasks, each of which is performed by one or more agents; these agents can have the form of automated systems or human agents. Agents that perform a part of a complex task are modelled as composed components. The functionality of a composed component is specified by the way in which it is composed from its subcomponents by means of a temporal declarative specification of task control and information exchange between its subcomponents. The functionality of a primitive component is specified in a declarative manner by a knowledge base. A number of reusable generic models for agents and tasks have been developed and may be reused: BDI agent, proactive & reactive, reflective agent, etc.

4.2.1.11 Dieng's methodology for specifying a co-operative system

The method of [Dieng, 1995] is inspired by knowledge engineering methodologies and cooperative expert systems. The idea is to engineer co-operative knowledge-based systems assisting collaborative work between experts and palliating the absence of one of them. The method relies on:
- *collective elicitation* protocol and analysis of associated expertise documents.
- *models of cognitive agents* to specify the knowledge-based systems that play the role of an expert

A model of a cognitive agent includes:
- *individual aspects*: this internal model includes general features (competence domain, high-level goals, expertise model, resources) and problem-specific features (state, intentions, history, information strategies). In parallel a KADS model of expertise (strategy layer, task layer, inference layer, ontological knowledge and assertional knowledge) is designed/
- *social aspects*: this external model describes the insertion and interactions of the agents in an organisation. It includes the description of co-operation modes, interaction points, communication languages, protocols, conflicts and associated resolution methods, organisational structure (if the agent is compound of sub-agents gathered by an organisational structure), model of other agents, social states and intentions, collective history.

The process followed is: (1) identification of the agents involved in the application, (2) modelling of the agents using models of a cognitive agent, (3) knowledge elicitation and knowledge modelling, (4) implementation of the agents (top-down or bottom-up for compound agents).

Of course some interactions occur between the different steps and elicitation (step 3) may reveal the need for new agents (step 1).

4.2.1.12 Elammari and Lalonde 's Agent-Oriented Methodology

[Elammari and Lalonde, 1999] propose a design process providing the following models:
- *System model*: a visual view of how the system works as a whole to accomplish some application related purpose. It is a macroscopic, system-oriented model to provide a means of both visualising the behaviour of systems of agents and defining how the behaviour will be achieved, at a level above such details.
- *Internal agent structure model*: agent internal structure in terms of goals, plans and beliefs together with a process to facilitate the discovery of agents needed along with their internal structure.
- *Relationships model*: agent dependencies and jurisdictional relationships with other agents together with a process to capture them.
- *Conversational model*: description of the conversations the agent can engage in together with a process to capture the conversational messages exchanged and facilitate the identification of conversational protocols used.
- *Contract model*: obligations and authorisations of agents together with a process to capture these commitments and any conditions or terms associated with them.

Authors insist on the necessity to provide systematic transitions: the design process must provide guidelines for model derivations and define traceability between the models.

The process consist of two phases:
- the *discovery phase* guides the identification of agents and their social behaviour to produce models that capture the social structure and behaviour of the system,
- the *definition phase* produces definitions of the behaviours, the entities that participate in exhibiting these behaviours and their interrelationships, as well as inter-agent conversations and commitments; these definitions can then be implemented.

4.2.1.13 GAIA

GAIA (the mother of life) is a top-down analysis influenced by object-oriented approaches [Wooldridge *et al.*, 1999]. The GAIA adopts a process of organisational design to build a multi-agent systems and borrows terminology and notation from object-oriented analysis and design and, in particular, from Fusion [Coleman *et al.*, 1994].

The methodology is intended to allow an analyst to go systematically from a statement of requirements to a design that is sufficiently detailed so that it can be implemented directly. The analyst moves from abstract to increasingly concrete concepts. Each successive move introduces greater implementation bias, and shrinks the space of possible systems that could be implemented to satisfy the original requirements statement. Analysis and design can be thought of as a process of developing increasingly detailed models of the system to be constructed as shown in Figure 33.

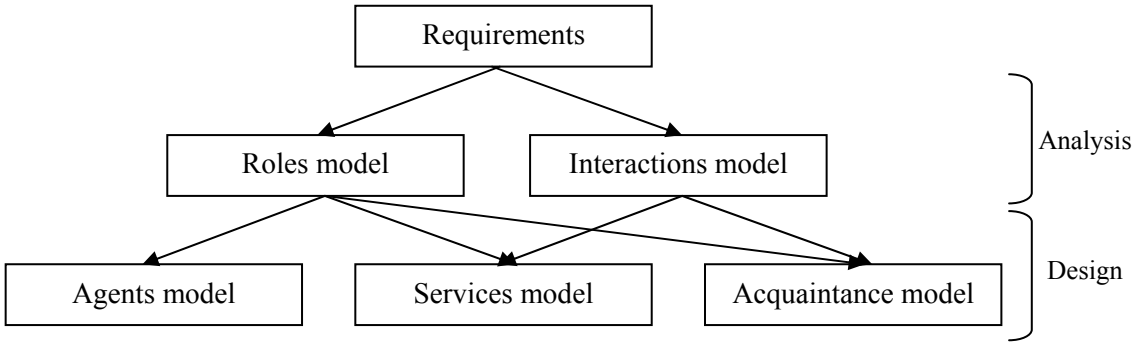


Figure 33 GAIA design process

The methodology is based on a set of abstract and concrete concepts:
- *Abstract Concepts* are used during analysis to conceptualise the system, but do not necessarily have any direct realisation within the system. They are: Roles, Permissions, Responsibilities, Protocols, Activities, Liveness properties, Safety properties.
- *Concrete Concepts* are used within the design process, and will typically have direct counterparts in the run-time system. They are: Agent Types, Services, Acquaintances.

The objective of the analysis stage is to capture the system organisation as a collection of roles that stand in certain relationships to one another and that take part in a set of systematic interactions with other roles. There is not a one-to-one mapping between roles and individual agents, and often an individual agent can take on many roles. A role is defined by four attributes: responsibilities, permissions, activities and protocols. Responsibilities determine functionalities and are divided into liveness properties (those states of affairs that an agent must bring about) and safety properties (invariants). Permissions are the "rights" associated with a role. Activities are computations associated with a role *i.e.,* private actions that an agent can carry out without interacting with other agents. Finally, protocols define the way that a role interacts with other roles.

Surprisingly, although the approach is organisation-centric, the organisation concept does not even appear as an abstract concept in the methodology. [Zambonelli *et al.*, 2000] stressed that point and added organisational rules, organisational structures, and organisational patterns as primary analysis and design concepts. Organisational rules express general, global requirements for the proper instantiation and execution of a system. An organisational structure defines the specific class of organisation and control regime to which the agents/roles have to conform in order for the whole MAS to work efficiently and according to its specified requirements. Organisational patterns express predefined and widely used organisational structures that can be reused from system to system.

4.2.1.14 Jonker *et al.*'s Design of Collaborative Information Agents

The methodological approach of [Jonker *et al.*, 2000] aims at providing a collaborative information agents in-depth analysis resulting in: the specification of requirements at different levels of the system; the specification of design structures; and a systematic verification.

It relies on two specification formal languages: a language to specify behavioural requirements and scenarios (TRL [Jonker *et al.*, 2000]) for systems of information agents, and a language to specify design descriptions (DESIRE *c.f. supra*). For both languages, informal, semi-formal and formal variants are assumed, to facilitate the step from informal to formal. Formal models are related in a formal manner defining when a design description satisfies a requirement or scenario specification.

The approach is top-down by refining requirements from the overall system to interactions and agents level. One of the objectives was to identify and classify a variety of instances of the different types of reusable patterns (requirement, design and proof), to build libraries of them, and to provide corresponding easy-to-use plug-in information agent components to the common user.

4.2.1.15 Kendall *et al.*'s methodology for developing agent systems for enterprise integration

This is an agent-oriented methodology for enterprise modelling [Kendall *et al.*, 1996]. It is a combination of object-oriented methodologies OOSE [Jacobson, 1992] and enterprise modelling methodologies IDEF [Fillion *et al.*, 1995] (Integration DEfinition for Function modelling) and CIMOSA [Kosanke, 1993] (Computer Integrated Manufacturing Open System Architecture).

The identified models are:
- *Functional model*: it describes the functions (inputs, outputs, mechanisms and control) using IDEF diagrams.
- *Use-case model*: it describes the actors involved in each function, using OOSE use case notation.
- *Dynamic model*: it describes the object interactions using event trace diagrams.

An additional model describes the agents and is itself a compound of:
- *Agent identification*: the actors of the use cases are identified as agents. The main functions of an agent are its goals and the possibilities described in the IDEF0 diagrams.
- *Co-ordination protocols*: they are described in state diagrams.
- *Plan invocation*: sequence diagrams extend event trace diagrams to include conditions for indicating when a plan is invoked.
- *Beliefs, sensors and effectors*: inputs of the functions are modelled as beliefs or obtained from objects via sensors, and achieved goals are modelled as changes to beliefs or outputs via effectors.

4.2.1.16 Kinny *et al.* 's methodology for systems of BDI agents

Also known as the AAII methodology [Kinny *et al.*, 1996] it is a top-down approach for systems of BDI agents [Rao and Georgeff, 1991], based on object-oriented methodologies enhanced with some agent-based concepts. The methodology aims at the construction of a set of models which, when fully elaborated, define an agent system specification. These models are divided into an internal and external perspective of multi-agent systems based on BDI architecture with an emphasis on roles, responsibilities, services and goals.

The external perspective presents a system-level view: the main components visible in this model are agents themselves. The perspective is used for the identification and description of the agents and their interactions. It is intended to define inheritance relationships between agent classes, and to identify the instances of these classes that will appear at run-time. It gives rise to two models:
- the *agent model* describing the hierarchical relationship between agents and relationships between concrete agents.
- the *interaction model* describing the responsibilities, services and interactions between agents and external systems.

The internal perspective focused on the internal BDI agent architecture and it naturally gives rise to three internal models:
- the *belief model* describing the beliefs of an agent about its environment.
- the *goal model* describing the goals of an agent and the events it can cause or respond to.
- the *plan model*, describing the plans an agent can use to achieve its goals.

An interesting aspect is that special attention was paid to the semantics of inheritance, aggregation and instantiation relationships amongst agent types and agent instances and their consequences on the state and behaviour of agents. The agent model is divided into an agent class model and an agent instance model. These two models define the agents and agent classes that can appear, and relate these classes to one another via inheritance, aggregation, and instantiation relations. Each agent class is assumed to have at least three attributes, for beliefs, desires, and intentions. The analyst is able to define how these attributes are overridden during inheritance. For example, it is assumed that by default, inherited intentions have less priority than those in sub-classes.

The analysis process is iterative:
1. external viewpoint - identification of the roles (functional, organisational, etc.) of the application domain in order to identify the agents and arrange them in an agent class hierarchy described using OMT like notation.
2. external viewpoint - identification of the responsibilities associated to each role are identified, together with the services provided and used to fulfil the responsibilities.
3. external viewpoint - identification of the necessary interactions for each service and both the speech-act and information content of every interaction.
4. external viewpoint - derive an agent instance model from the collected information.
5. internal viewpoint - analysis of the different plans for achieving each goal: determine the plans that may be used to achieve a goal and the context conditions under which each plan is appropriate. The plans for responding to an event or achieving a goal are represented using a graphical notation similar to state-charts.
6. internal viewpoint - analysis of the beliefs mobilised bay the agents for each plan and goal.

4.2.1.17 MAS-CommonKADS

MAS-CommonKADS [Iglesias *et al.*, 1998] extends CommonKADS [Breuker and Van de Velde, 1994] for multiagent systems modelling, adding techniques from object oriented methodologies such as Object Modelling Technique (OMT) [Rumbaugh *et al.*,1991], Object Oriented Software Engineering (OOSE) [Jacobson, 1992], Responsibility Driving Design (RDD) [Wirfs-Brock *et al.*, 1990] and from protocol engineering for describing the agent protocols, such as Specification and Description Language [ITU-T, 1994] and Message Sequence Charts [Rupolph *et al.*, 1996].

This methodology starts with a conceptualisation phase from which it defines a first description of the system from the user point of view. Then, it uses a set of models for the analysis and design:

- *Agent model*: it describes the main characteristics of the agents, including reasoning capabilities, skills (sensors/effectors), services, goals, etc. Several techniques are proposed for agent identification, such as analysis of the actors of the conceptualisation phase, syntactic analysis of the problem statement, application of heuristics for agent identification, reuse of components developed previously or usage of CRC cards, etc.
- *Task model*: describes the tasks (goals) carried out by agents, and task decomposition, using textual templates and diagrams.
- *Expertise model*: describes the knowledge needed by the agents to carry out the tasks. The knowledge structure follows the KADS approach, and distinguishes domain, task, inference and problem solving knowledge. Several instances of this model are developed for modelling the inferences on the domain, on the agent itself and on the rest of agents. A distinction is made between autonomous problem solving methods that can be directly carried out by the agent itself and co-operative problem solving methods, that are carried out by the agent in co-operation with other agents.
- *Co-ordination model*: describes the conversations between agents, that is, their interactions, protocols and required capabilities. The development of the model defines two milestones. The first milestone is intended to identify the conversations and the interactions. The second milestone is intended to improve these conversations with more flexible protocols such as negotiation and identification of groups and coalitions. The interactions are modelled using the formal description techniques MSC (Message Sequence Charts) and SDL (Specification and Description Language).
- *Organisation model*: describes the *organisation in which the MAS is going to be introduced and the organisation of the agent society*. The description of the multi-agent society uses an extension of the object model of OMT, and describes the agent hierarchy, the relationship between the agents and their environment, and the agent society structure.
- *Communication model*: details the human-software agent interactions, and the human factors for developing these user interfaces.

A *design model* then collects the previous models and is subdivided into three sub-models:

- *application design*: composition or decomposition of the agents of the analysis, according to pragmatic criteria and selection of the most suitable agent architecture for each agent.
- *architecture design*: designing of the relevant aspects of the agent network: required network, knowledge and telematic facilities etc.,
- *platform design*: selection of the agent development platform for each agent architecture.

For each model, the methodology defines the constituents (entities to be modelled) and the relationships between the constituents. The methodology defines a textual template for describing each constituent and a set of activities for building each model, based on the development state of each constituent (empty, identified, described or validated). These activities facilitate the management of the project.

4.2.1.18 MASB

MASB stands for Multi-Agent Scenario-Based Method [Moulin and Brassard 1996]. It was proposed in the field of Computer Supported Collaborative Work. The analysis phase consists of the following activities:

- *Scenario description*: textual identification and description of the typical scenarios with the roles played by human actors, software agents, objects and the environment.
- *Role description*: description of the agent roles using behaviour diagrams. It describes the processes, the relevant information and the interactions between the agents.
- *Data and world conceptual modelling*: modelling of the data and knowledge used by the agent using entity-relationship diagrams or object-oriented diagrams and entity lifecycle diagrams.
- *System-user interaction modelling*: simulation and definition of different suitable interfaces for human-machine interaction in each scenario.

Then, the design phase consists of the following activities:

- *MAS architecture and scenario description*: select the scenarios to be implemented and the roles played by the agents in these scenarios.
- *Object modelling*: refine the world modelling of the analysis, defining hierarchies, attributes and procedures.
- *Agent modelling*: specify of the agent internal structure. A graphical notation is proposed for describing the decision process of a agent, taking into account beliefs, plans, goals and interactions.
- *Conversation modelling*: specify the interactions (not detailed).
- *Design validation*: validate the architecture (not detailed).

4.2.1.19 MESSAGE

MESSAGE is an agent oriented software engineering methodology [Caire *et al.*, 2001]. This methodology derives from the Rational Unified Process and borrows some terminology and notation from UML. The following diagram types are introduced: organisation, goal, task, delegation, workflow, interaction and domain. All are extensions of UML class diagrams, except for the task diagram, which extends the UML activity diagram. The authors also use schemas to textually describe the concepts.

MESSAGE foundation concepts are: Agent, Organisation, Role, Resource, Task, Interactions and Interaction protocols, Goals, Information, Message. These concepts are mobilised in several overlapping views:

- *Organisation view* (Agents, Organisations, Roles, Resources and relationships between them),
- *Goal/Task view* (Goals, Tasks, Situations and the dependencies among them),
- *Agent/Role view*,
- *Interaction view* (for each interaction among agents/roles),
- *Domain view* (domain specific concepts).

MESSAGE methodology is based on three phases: requirements, multi-agent system analysis and multi-agent system design. It also gives some guidelines for the implementation phase.

For the requirements phase, MESSAGE does not define a new method, but recommends the use of existing requirement methods for establishing the requirements for a multi-agent system.

The analysis models are produced by stepwise refinement choosing among the following strategies: organisation-centred top-down identification; agent-centred bottom-up; goal/task decomposition based on functional decomposition. A combination of these refinement strategies with loop-backs among them is also advocated.

4.2.1.20 Role card model for agents

This is just one model for agent roles [Kendall, 1999], but I find it important since the role concept is at the turning point between macro-level analysis and micro-level analysis in nearly all the methodologies presented here.

Role models are relatively new concepts in object oriented software engineering: they emphasise patterns of interaction, and Kendall explains how they can be used to facilitate agent system analysis and design.

Role models emphasise how entities interact with each other, focusing on their position and responsibilities within an overall structure or system. It is described in terms of patterns of interaction that can be instantiated, generalised, specialised, and aggregated into compound models, thus promoting reuse. They are similar to UML collaboration diagrams with the addition of being reusable and extensible. Facets of an agent role are:

- *role model*: the name and context of the role.
- *responsibilities*: the services, tasks, goals, obligations and interdictions associated to the role.
- *collaborators*: the other roles it interacts with.
- *external interfaces*: the access it has to services external to the multi-agent system.
- *relationships* to other roles: aggregation, specialisation, generalisation, sequences or other roles.
- *expertise*: the ontology, inferencing, problem solving methods, knowledge, etc. the role has.
- *co-ordination and negotiation*: protocol, conflict resolution, knowledge of why other roles are related, permissions.
- *other* resources, e.g.: learning/ adaptability.

When implementing, some kinds of behaviour or functionality cross cut or are orthogonal to the agent types or roles and they are not easily modularised into a separate components. Kendall suggested and explain how to use aspect-oriented design in these cases.

4.2.1.21 Verharen's methodology for co-operative information agents design

The corresponding Ph.D. [Verharen, 1997] is called language-action perspective on the design of co-operative information Agents. It includes a methodology from a business-process perspective, with the following four models:

- *Task model*: specifies the tasks and their decomposition into sub-tasks, using task diagrams.
- *Authorisation model*: describes the authorised communication and obligations between the organisation and the environment and the internal communications using authorisation diagrams. It is used both for describing the current situation and for redesigning it to improve the business processes.
- *Communication model*: refines the previous model describing in detail the contracts between the agents. The model uses Petri-nets and transaction diagrams that describe the relationship between speech-acts and goals.
- *Universe of discourse model*: concerns the modelling of the content of the message exchanged between the agents, using object-oriented techniques. It is close to the ontological considerations.

4.2.1.22 Vowels

Vowels [Demazeau, 1995] is an approach where multi-agent systems are envisaged through four models corresponding to four vowels:

A	model of the **A**gent internal active structure from simple automata to complex knowledge-based systems.
E	model of the **E**nvironment in which the agents are situated, from spatial representation to virtual abstract worlds.
I	model of the infrastructures, languages and protocols for **I**nteractions between agents, from simple physical interactions to communications based on speech acts.
O	model of the **O**rganisation that structures the agents in groups and the forms of this social structures.

Based on the Vowels models there are three principles:
- The declarative principle: **MAS = Agents + Environment + Interactions + Organisation** *i.e.,* a MAS must be described by instantiating these four models.
- The functional principle: **Function(MAS) = Σ Function (Agent$_i$) + Collective Function** *i.e.,* the functions that will be ensured by the multi-agent system are enclosing the ones of the agents enhanced by the interactions between the agents and the environment or the other agents.
- The recursive principle: **MAS* = MAS** *i.e.,* societies of agents should be able to be considered as coarser agents at a higher level of abstraction and should be handled as such. It is close to the holonic view of agents.

The analysis goes on developing each dimension; platforms and toolkits have been proposed to support this approach.

4.2.1.23 Z specifications for agents

This is one of the most formal approaches based on an agent specification framework using the Z language [Luck *et al.*, 1997]. The Z language is geared towards the specification of operation-based, functional systems. The emphasis is placed on the notion of agents acting for another, rather than on agents as rational systems. The types of agents that the approach allows to develop are thus inherently different from "rational" agents.

The authors define a four-level hierarchy of the entities that can exist in an agent-based system. The formal definitions of an entity rely on inheriting the properties of lower-level ones.
- The first level is the one of *entities*, which are inanimate objects that have attributes.
- The second level is the one of *objects* that are entities with capabilities.
- The third level is the one of *agents* that are active objects with goals.
- The fourth level is the one of *autonomous agents* that are agents with motivations.

The whole framework descriptions and verifications are completely formal.

4.2.1.24 Remarks on the state of the art of methodologies

The above methods span different levels of formality, support, validation, generality/specificity, availability etc. There are broadly four main categories of contributions, and only sometimes several of them are addressed in a methodology:
- *design rationale* for collaborative design of agent systems with natural language guidelines for designers,
- *design notations* for documenting design productions,
- *design formal specifications* for validation and hopefully for generating at least the shallow architectures specified,
- *design reusability* through component-based or alike approaches.

The general remarks I would make in the perspective of a complete methodology are:
- Few comparisons and nearly no coupling exist between the too many methodologies.
- Few methodologies really try to unify the design process.
- Few methodologies model the human organisation in which the system will be deployed.
- Few methodologies acknowledge the interface importance and the fact that both agent-agent and human-agent interactions must be taken into account.
- Few methodologies consider a complete paradigm with both agents and objects taking into account their ontological differences. The 'all agents' tendency is a mistake; a good agent-oriented design methodology must assist designers in the identification and characterisation of entities as either agents or objects. [Wooldridge and Ciancarini, 2000]

One of the constant is that there is always at least two types of models [Iglesias *et al.*, 1999]: the *individual agent models* (where the characteristics of each agent type are described, including skills, sensors and effectors, reasoning capabilities, tasks, etc.) and *the society models* (where the relationships and interactions between the agents are described). Les Gasser [1991] termed them the micro-aspects and the macro-aspects of the multi-agent systems, the latter emphasises the societal aspects of agents over the internal individual aspects agents that are developed in the former. It is also clear that the roles and their interactions are a turning point between these two models.

In fact, these notions are here since the pioneer works like MACE [Gasser *et al.*,1987] where the entity 'organisation' reifies the abstraction of a set of agents acting together and enables agents to consider it as another entity playing a role. Thus, these few shared notions that are at the heart of the paradigm, should be used as a start point for the unification of the methodologies.

Also, initiatives like the comparison work done in MUCCMAS (MUltidimensional framework of Criteria for the comparison of Multi-Agent Systems methodologies) [Sabas *et al.*, 2002] give interesting dimensions characterising methodologies: they can be used to compare and merge or complete the actual contributions. I slightly reformulated and modified these dimensions here:

1. Design dimension:
   - what is the design process? cascade process, V-shape process, spiral process, incremental process, prototypical process, nabla, etc.
   - what is the modelling process? top-down, bottom-up, middle-out, evolving, hybrid, etc.
   - what are the stages of the process? analysis, modelling, specification, design, validation, verification, ergonomic evaluation, etc.
   - what is the implication of user? degree of implication, design stages where the implication occur, means of communications, etc.
   - what are the reuse opportunities?
   - what are the supporting tools?
2. Representation dimension:
   - what are the abstractions proposed? level of abstraction, core notions and their relations (aggregation, subsumption, instantiation, etc.)

- what are the supporting formalisms? notations, diagrams, rules etc.
- what are the relations between the models? sequence, derivation, interactions, overlaps, etc.
- what is the quality of the models? coverage, cohesion, complexity, etc.

3. Agent dimension:
   - what is the nature of the population of agents? homogeneous, heterogeneous,
   - what are the agent types? interface, information, matchmaker, etc.
   - what are the intrinsic agent abilities? learning, knowledge-based, deliberative, etc.
   - what are the agent behaviours? internal structure, decision mechanisms, etc.
4. Organisation dimension:
   - what are the boundaries? open/closed, centralised/distributed, etc.
   - what are the structures? groups, relations (hierachies, peer-to-peer, aggregation etc.), etc.
5. Environmental dimension:
   - what is the application domain? web mining, simulation, knowledge management, etc.
   - what is the environment made of? digital world (data, objects, etc.), real world (signals, physical entities, etc.)
   - what are the environmental rules? static, stable, dynamic, active, determinist, observable, etc.
6. Co-operation dimension:
   - what are the actors of the co-operation? agent-agent co-operation, human-agent co-operation, etc.
   - what are the communication modes? direct, indirect, synchronous, asynchronous, etc.
   - what are the communication means? environment modification, signals, languages, etc.
   - what are the co-operation patterns? negotiation, delegation, task distribution, etc.
   - what are the interaction rules? static, dynamic, protocols, conflicts resolution, etc.
7. Technology dimension:
   - what are the processing means? batch, iterative, parallel, distributed, etc.
   - what are the interfaces? device characteristics (sound, screen, etc.), user characteristics, etc.
   - what is the implementation language abstraction? objects, components, agents, etc.
   - what are the implementation supporting tool? platforms, editors, debuggers, etc.

Finally, it is clear that the gap between formal approaches and informal approaches is still very large, but that is not specific to the domain of agent-based design.

4.2.2 The organisational design axis and role turning-point

I suppose society is wonderfully delightful.
To be in it is merely a bore.
But to be out of it is simply a tragedy.
— Oscar Wilde

We saw that one of the constant and oldest design axis is the organisational dimension. Thus it is the one I shall favour in the rest of this work. It is usually split in two levels, but the whole process followed, be it in a top-down or bottom-up fashion, is usually composition or decomposition all along the axis. The two main conceptual levels are the internal agent model and the collective agent models; these models being possibly nested in an holonic fashion or reifying the notion of group to consider it as an addressable entity. The intertwined concepts of individual agent and multi-agent systems are symptomatic of the two perspectives at play in the paradigm. When envisaging a software solution in a multi-agent perspective, the designer is perpetually oscillating between:
- the *macroscopic level* of the multi-agent system (the system as a whole), that poses the problems of engineering the interactions and the organisation within the agent society to get, from the overall point of view of the system, the functionality matching the user's requirements.
- the *microscopic level* of the multi-agent system (agents as individuals), that poses the problems of engineering the behaviours and providing the right individual skills that will benefit to the whole system.

The whole problem is to match the societal-level requirements where agents are atomic entities that communicate and the individual agent implementation where the social global view is out of reach.

The adoption of an organisational perspective suggests that there will be a notion of structure and therefore, the first idea is to look for structural patterns in organisations. [Sycara, 1998b] gives, for instance, the following examples of organisations:
- *Hierarchy*: the authority for decision making and control is concentrated in an agent or a group of agents at each level in the hierarchy. Interaction is through vertical communication from superior to subordinate agents, and vice versa. Superior agents exercise the control.
- *Community of experts*: this is a flat organisation, where each agent is specialised in some particular area. Agents co-ordinate though mutual adjustment of their solutions so that overall coherence can be achieved.
- *Market*: control is distributed to the agents that compete for tasks or resources through bidding and contractual mechanisms. Bidding is done through one measure (e.g. price, reward, etc.) which is used to value offers. Agents co-ordinate through mutual adjustment of bids.
- *Scientific community*: agents locally construct solutions to problems and communicate them to other agents that can test, challenge, and refine the solution.

These examples use notions such as "hierarchy", "community", "contractual mechanisms", "superior agent", "agent that compete", etc. For me, they suggest the following important aspects to be considered following the organisational dimension:
- *groups* with *structures* (e.g. hierarchy) defined in terms of ***roles*** (played by groups or individual) and *relationships*
- *interactions* defined in terms of *rules* (e.g. protocols) and ***roles*** involved (played by groups or individual)
- *profiles* and *behaviours* of an *agent type* that plays a ***role***.

In this view, roles seem to be an entity involved in all the aspects. They are also at the design junction between the micro-level of agents and the macro-level of the multi-agent system. Their introduction is driven by functional requirements and their implementation is subject to the toolbox of technical abilities available.

Roles may be played by groups, thus it is not surprising if roles are found at every level of the organisational dimension and it is also foreseeable that a role may be compound of other roles.

Roles are more or less generic e.g. in a generic protocol we may have an emitter, a receptor, etc. while in a middle-agent group, we may have a matchmaker, a broker, etc.

By creating a junction between the societal and individual levels, roles are a turning point in the design. They are specification links and carriers of constraints between the various models specifying the different aspects of a system.

4.3 Agent communication

The architecture design reveals the importance of roles. But roles to participate to a collective behaviour need to interact. As I positioned this work in the domain of symbolic artificial intelligence, these interactions will take place at the knowledge level and therefore require an appropriate agent communication language. In this section, I shall look at the needs for syntactic and semantic grounding, the needs for communication architectures conducive to co-operation and finally, briefly introduce two languages commonly used.

4.3.1 Communication language: the need for syntactic and semantic grounding

The very first problem that appears when considering the society level is the communication language. In order to perform their tasks effectively, agents depend heavily on expressive communication with other agents, to perform their requests, to propagate their information capabilities, to negotiate with other agents etc. [Papazoglou and Heuvel, 1999]

Problems of communication languages are of two types [Genesereth and Ketchpel, 1994]:
- *Inconsistencies* in the use of syntax or vocabulary. One program may use a word or expression to mean one thing while another program uses the same word or expression to mean something entirely different.
- *Incompatibilities* between different programs using different words or expressions to say the same thing. In fact, we see here a direct field of application for ontologies and knowledge modelling languages.

The first challenge of collaboration among heterogeneous and autonomous agents is that of mutual understanding [Klusch, 1999a]. There exists basic frameworks for distributed computing in heterogeneous environments like CORBA, DCOM, Java RMI. They provide interface description language and services that allow distributed objects to be defined, located and invoked. But complex co-operation and heterogeneity of information require knowledge level interactions among agents; this goes beyond the method invocation paradigm and it requires communication languages providing a structural and semantic grounding.

″Additional efforts are required to achieve interoperability among the potentially heterogeneous information agents and systems at all levels in uncertain environments without any global control (...). Such efforts include the generation and use of ontologies and communication conventions (...), non-proprietary languages for knowledge interchange, and methods for co-ordinating collaborative work among different agents (...).″[Klusch, 99a]

It is interesting to notice that this need of a common language meets the W3C initiative of XML and RDF languages exposed before. The use of standardised non proprietary markup languages for multi-agent systems would be wise move to enable to quickly make agents smarter through semantic annotations and machine understandable structuring of documents. "Our hope to achieve open multi-agent applications (on the WWW say) without having addressed the ontology problem is rather naïve." [Nwana and Ndumu, 1999]

Agent languages can be divided in two approaches [Genesereth and Ketchpel, 1994]: the *procedural approach* is based on the exchange of procedural directives using scripting languages (e.g. TCL, Apple Events, Telescript) to transmit entire programs, implementing delayed or persistent goals of various sorts; the *declarative approach* is based on the exchange of declarative statements (definitions, assumptions, etc.) and it must be sufficiently expressive to communicate information of widely varying sorts.

4.3.2 Communication protocols: the need for interaction rules

However, communication typically consists in more than sending an isolated message even with the correct syntax and semantic. Communication usually involves several messages that form a dialogue. These dialogues follow patterns or policies that ensure a coherent exchange and this coherence can be reached only if the two participants are explicitly aware of the particular pattern in which they are engaged. [Purvis *et al.*, 2000]

Thus above the syntactic and semantic aspects is the pragmatic aspect of communication protocols *i.e.,* normative social behaviour rules. The protocols that are needed for agents, go beyond the client-server example and describe in terms of the high level of communication languages, the sequences of messages that can occur and their meaning in order to ensure the coherence required to initiate, carry out and terminate a co-operation. Protocols have a normative force that requires that the agents behave felicitously [Singh and Huhns, 1999], but in exchange they ensure a known and agreed set of possible events and outcomes. "A protocol can be viewed as an institutionalised pattern of interaction. That is, a pattern of interaction that has been formally defined and abstracted away from any particular sequence of execution steps. Viewing interactions in this way means that attention is focused on the essential nature and purpose of the interaction, rather than on the precise ordering of particular message exchanges. "[Wooldridge *et al.*, 1999]

[Genesereth and Ketchpel, 1994] distinguish between *direct communication* (*i.e.,* agents handle their own co-ordination) and *assisted co-ordination* (*i.e.,* agents rely on special system programs to achieve co-ordination). Assisted co-ordination tends to be a reminiscence of centralised solutions. And most of the protocols proposed now are fully distributed between the agents playing the roles involved in the protocol.

The protocols define communication patterns that can be used to set up relationships that define a special organisational pattern. For instance, one of the most well-known protocols is the *contract-net* [Davis and Smith, 1983]: agents in need of services distribute calls for proposals (CfP) to other agents. The recipients of these messages evaluate those requests and submit bids to the originating agents. The originators use these bids to decide which agents to choose and then award contracts to those agents. Contract-net can be used for instance in a task allocation problem to decide which is the best agent to be assigned the responsibility of solving a problem. In institutionalising interactions, the protocol dynamically assigns two roles: manager or contractor. Doing so, the application of a protocol generates an organisational structure.

Unfortunately, while the fact of being aware of the communication pattern is vital for a system to work properly, no conversation policy standards have emerged yet.

> "The complex social interactions involved in human negotiation, co-operation, and teamwork are among the most difficult to adopt by software agents. Research in modelling such interactions and the strategies they entail, how to design rules and semantics for conversations based on the semantics of speech acts, negotiation protocols, auctions, and so on, continues. It is inspired especially by related work from CSCW and cognitive and social sciences" [Klusch, 1999a].

4.3.3 Communication and content languages

A number of inter-agent communication languages have been proposed, but to date there are only two candidates at a standardised agent communication language (ACL) that emerged from the knowledge modelling and linguistic studies: KQML (Knowledge Query and Manipulation Language) [Finin *et al.*, 1994] and FIPA ACL [FIPA]. Up to now they are the only ACL that are widely implemented and used.

These languages are based on speech act theory performatives [Searle, 1969], wherein the speaker's intent of the effects of a message on the hearer is communicated by specifying the type of the message, e.g. ask, tell, or achieve message types. The dictionary of philosophy of Kemerling explains very pedagogically these notions:

- *Speech acts*: The complex group of things we typically perform when speaking. J. L. Austin famously distinguished the simple locutionary act of saying something meaningful, the force of the illocutionary act of employing this language for some purpose, and the further perlocutionary act of having an actual effect on those who hear the utterance. Thus, for example, in saying (locution) to a friend, "That's an ugly necktie," I may also insult him (illocution) and persuade him to dress differently (perlocution).
- *Locutionary act*: The simple speech act of generating sounds that are linked together by grammatical conventions so as to say something meaningful. Among speakers of English, for example, "It is raining" performs the locutionary act of saying that it is raining, as "Grablistrod zetagflx dapu" would not.
- *Illocutionary act*: The speech act of doing something else—offering advice or taking a vow, for example—in the process of uttering meaningful language. Thus, for example, in saying "I shall repay you this money next week," one typically performs the illocutionary act of making a promise.
- *Perlocutionary act*: The speech act of having an effect on those who hear a meaningful utterance. By telling a ghost story late at night, for example, one may accomplish the cruel perlocutionary act of frightening a child.

It is based on these philosophical distinctions that the actual ACL are defined: speech act ontological primitives are standardised in a language to be used to type the messages of the agents. Thus a first use of ontologies is to capture these speech acts to use them as message typing primitives. As an example, FIPA ACL standardised the primitives given in Table 7.

KQML and FIPA ACL concentrate on the ontologies for the message envelope. The content part is deliberately kept unspecified. But, although such performatives can characterise message types, efficient languages to express message content that allows agents to understand each other are still needed to encode the content of a message. KIF (Knowledge Interchange Format) [Genesereth and Fikes, 1992] is usually used as content language for KQML message language. FIPA SL was proposed by FIPA, but any ontology and knowledge representation language from the knowledge engineering community and lately the W3C can provide content language. FIPA already considered the use of RDF(S).

Accept Proposal	The action of accepting a previously submitted proposal to perform an action.
Agree	The action of agreeing to perform some action, possibly in the future.
Cancel	The action of one agent informing another agent that the first agent no longer has the intention that the second agent performs some action.
Call for Proposal	The action of calling for proposals to perform a given action.
Confirm	The sender informs the receiver that a given proposition is true, where the receiver is known to be uncertain about the proposition.
Disconfirm	The sender informs the receiver that a given proposition is false, where the receiver is known to believe, or believe it likely that, the proposition is true.
Failure	The action of telling another agent that an action was attempted, but the attempt failed.
Failure	The action of telling another agent that an action was attempted, but the attempt failed.
Inform If	A macro action for the agent of the action to inform the recipient whether or not a proposition is true.
Inform Ref	A macro action for sender to inform the receiver the object which corresponds to a descriptor, for example, a name.
Not Understood	The sender of the act (for example, *i*) informs the receiver (for example, *j*) that it perceived that *j* performed some action, but that *i* did not understand what *j* just did. A particular common case is that *i* tells *j* that *i* did not understand the message that *j* has just sent to *i*.
Propagate	The sender intends that the receiver treats the embedded message as sent directly to the receiver, and wants the receiver to identify the agents denoted by the given descriptor and send the received propagate message to them.
Propose	The action of submitting a proposal to perform a certain action, given certain preconditions.
Proxy	The sender wants the receiver to select target agents denoted by a given description and to send an embedded message to them.
Query If	The action of asking another agent whether or not a given proposition is true.
Query Ref	The action of asking another agent for the object referred to by an referential expression.
Refuse	The action of refusing to perform a given action, and explaining the reason for the refusal.
Reject Proposal	The action of rejecting a proposal to perform some action during a negotiation.
Request	The sender requests the receiver to perform some action. One important class of uses of the request act is to request the receiver to perform another communicative act.
Request When	The sender wants the receiver to perform some action when some given proposition becomes true.
Request Whenever	The sender wants the receiver to perform some action as soon as some proposition becomes true and thereafter each time the proposition becomes true again.
Subscribe	The act of requesting a persistent intention to notify the sender of the value of a reference, and to notify again whenever the object identified by the reference changes.

Table 7 Extract of the FIPA communicative act library [FIPA]

The second interesting initiative of FIPA is to propose a library of protocols and an associated description notation. A list of some of the most used FIPA protocols is given in Table 8, they use the speech acts of Table 7.

Contract Net Interaction Protocol	minor modification of the original contract net [Davis and Smith, 1983] pattern in that it adds rejection and confirmation communicative acts. In the contract net, one agent takes the role of manager which wishes to have some task performed by one or more other agents and further wishes to optimise a function that characterises the task. This characteristic is commonly expressed as the price, in some domain specific way, but could also be soonest time to completion, fair distribution of tasks, etc.
Dutch Auction Interaction Protocol	the auctioneer attempts to find the market price for a good by starting bidding at a price much higher than the expected market value, then progressively reducing the price until one of the buyers accepts the price. The rate of reduction of the price is up to the auctioneer and they usually have a reserve price below which not to go. If the auction reduces the price to the reserve price with no buyers, then the auction terminates.
English Auction Interaction Protocol	the auctioneer seeks to find the market price of a good by initially proposing a price below that of the supposed market value and then, gradually raising the price. Each time the price is announced, the auctioneer waits to see if any buyers will signal their willingness to pay the proposed price. As soon as one buyer indicates that it will accept the price, the auctioneer issues a new call for bids with an incremented price. The auction continues until no buyers are prepared to pay the proposed price, at which point the auction ends. If the last price that was accepted by a buyer exceeds the auctioneer's (privately known) reservation price, the good is sold to that buyer for the agreed price. If the last accepted price is less than the reservation price, the good is not sold
Iterated Contract Net Interaction Protocol	an extension of the basic contract net IP. It differs by allowing multi-round iterative bidding. The manager issues the initial call for proposals with the cfp act. The contractors then answer with their bids as propose acts and the manager may then accept one or more of the bids, rejecting the others, or may iterate the process by issuing a revised cfp. The intent is that the manager seeks to get better bids from the contractors by modifying the call and requesting new (equivalently, revised) bids. The process terminates when the manager refuses all proposals and does not issue a new cfp, accepts one or more of the bids or the contractors all refuse to bid.
Propose Interaction Protocol	an initiator agent proposes to the receiving agents that the initiator will do the actions described in the proposed communicative act when the receiving agents accept this proposal. Completion of this IP with an accept-proposal act would typically be followed by the performance of the proposed action and then the return of a status response.
Query Interaction Protocol	the receiving agent is requested to perform some kind of inform action. Requesting to inform is a query, and there are two query-acts: query-if and query-ref and either act may be used to initiate this protocol. In either case, an inform is used in response, although the content of the inform given in response to a query-ref would be a referring expression.
Request Interaction Protocol	simply allows one agent to request another to perform some action and the receiving agent to perform the action or reply, in some way, that it cannot.
Request When Interaction Protocol	provides a framework for the request-when communicative act. The initiator uses the request-when action to request that the participant do some action once a given precondition becomes true. If the requested agent understands the request and does not initially refuse, it will agree and wait until the precondition occurs. Then, it will attempt to perform the action and notify the requester accordingly. If after the initial agreement the participant is no longer able to perform the action, it will send a refuse action to the initiator.
Subscribe Interaction Protocol	an agent requests to be notified whenever a condition specified in the subscription message becomes true

Table 8 Extract of the FIPA protocol library [FIPA]

Finally, the following schema in Figure 34 shows an example of the protocol diagram notation for the English Auction protocol of FIPA..

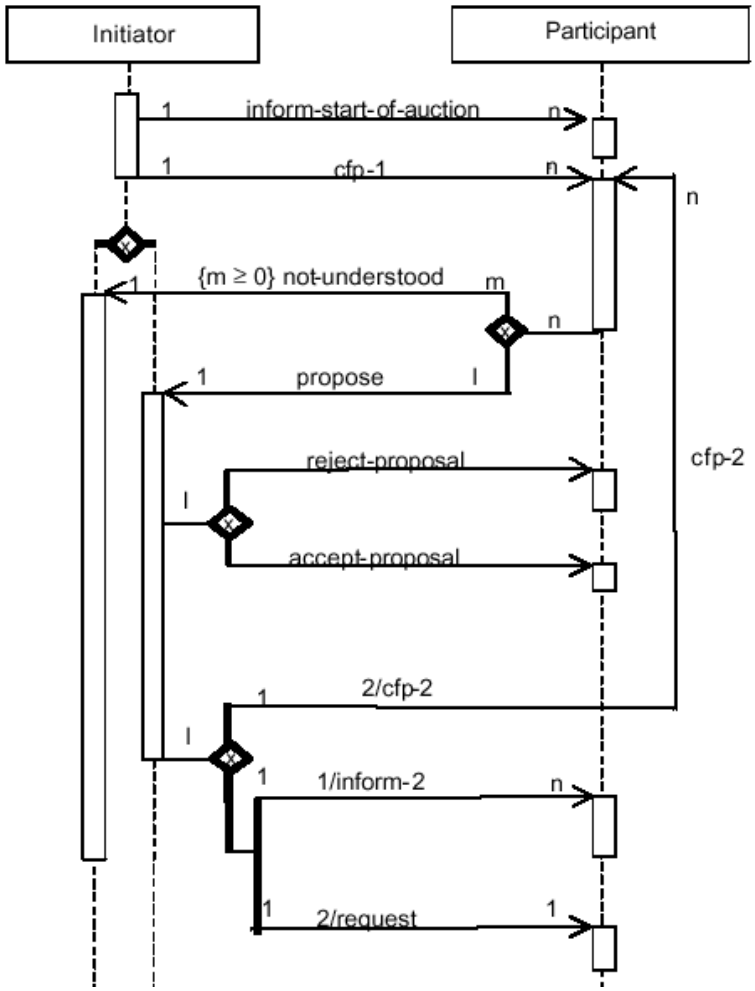


Figure 34 English auction protocol diagram [FIPA]

4.4 Survey of multi-agent information systems

This part will provide a survey on some co-operative information relevant to this work; I shall use it to position my work in the research and development area. There would be no point in giving here a complete overview of multi-agent systems which are now covering a wide range of application domains, and this part is only focusing on the state of the art of Multi-Agent Information Systems.

4.4.1 Some existing co-operative or multi-agent information systems

This part will give a non-comprehensive set of examples of multi-agent information systems. It aims to both exemplify the existence of multi-agent information systems research and development field and give some of the initiatives that inspired my work.

[Finin *et al.*, 1998] classifies some examples of the agent-based information systems as follows:

- *collaborative filtering*: recommendation to a person based on the preference of similar users; e.g. recommending persons (Yenta, ReferralWeb), recommending movies (EachMovie, Morse, RARE, MovieCritic), recommending music (Firefly, Similarities Engine, Tunes), recommending readings (Firefly, Fab, Phoaks)
- *adaptive interfaces*:
  - proactive indexing or sorting e.g.: Letizia (browses in breadth in parallel with the user and suggests other pages), RemembranceAgent (indexes personal files and emails, suggest documents based on task analysis)
  - proactive push of document (e.g. via e-mail): Backweb, Marimba, Pointcast, SIFT, TopicAGENTs
- *large heterogeneous distributed information systems*: CARROT (using KQML language and existing information retrieval engine based on N-Gram), InfoSleuth, Retsina, SAIRE
- *knowbot* that provide a single query language:
  - Shopbots used to assist online shopping e.g. Netbot Jango, Shopbot,
  - Meta-Search exploiting search engines to provide better search capabilities e.g: All-in-one, Fastfind, Metacrawler, Metasearch, Profusion, Savvysearch, WebCompass.

I shall detail some chosen examples to illustrate the types of functionalities that may be envisaged.

4.4.1.1 Enterprise Integration Information Agents

In [Barbuceanu and Fox, 1994], information agents are a component of the information infrastructure supporting collaborative computing environments. They identified some agents and the functions they can play in a co-operative information system environment. Their architecture is composed of:

- a description logics engine for reasoning.
- an agent communication language based on KQML/KIF.
- generic information distribution service allowing agents to describe their interests and automatically receive information that matches these interests.
- general organisation modelling framework (organisations, agents that compose them, agents' roles, goals, positions and communication links, agents' credibility in organisations)
- a language for explicitly describing co-operation models as structured conversations among agents, expressed as rules and conversation objects, and that can be initiated, interrupted and resumed using a nested conversation structure.
- conflict management model to resolve contradictions among the beliefs of agents. It is based on agents' credibility in the organisation and the costs of retracting beliefs.

It was used to build information agents in the framework of the Integrated Supply Chain.

4.4.1.2 SAIRE

SAIRE stands for Scalable Agent-based Information Retrieval Engine [Odublyi *et al.*, 1997], it is a multi-agent search engine employing intelligent software agents, natural language understanding, and conceptual search techniques to support public access to Earth and Space Science data over the Internet. SAIRE provides an integrated user interface to distributed data sources maintained by NASA and NOAA thereby hiding complex information retrieval protocols from users.

SAIRE has demonstrated the feasibility of a multi-agent architecture, multi-agent collaboration, and communication between people and agents to support intelligent information access and retrieval.

Authors noticed that to promote case of use, an information retrieval system should provide users with a multi-modal interface. In SAIRE, users are then able to submit requests in a preferred input format (e.g. natural language, graphical menus, commands, etc.). SAIRE's Natural Language Parser (nl-parser) takes a natural language statement as input and works to output a frame containing actions and important concepts embedded in the natural language input.

The introduction of natural language processing is an example of the influence of long-existing area inside multi-agent information systems. But for interfacing the user, SAIRES also uses user modelling. Each user specifies a group preference at login. The user groups are science, general science, or non-science. Individual users can then be modelled by customising an instance of their profile from the general class. Beliefs regarding each user can be modified by reasoning non-monotonically with new information inferred from SAIRE's operational environment. This supports dynamic adaptation to user preferences. User preferences are stored in individual user dictionaries and retrieved as needed.

4.4.1.3 NZDIS

NZDIS stands for New Zealand Distributed Information Systems [Purvis *et al.*, 2000]. It aims at developing a system to facilitate connecting the many disparate and distributed databases across New Zealand. The distributed information systems architecture is agent-based. It relies on ontologies and the representation formalism used in the NZDIS project is a subset of the OMG's Unified Modelling Language, together with its associated Object Constraint Language. The Object Query Language (OQL) is the query language

The NZDIS architecture is designed to provide an open agent-based environment for the integration of disparate sources of information. The agent types developed are:

- *User Agent* to formulate queries.
- *Ontology Agent* to provide relevant ontologies.
- *Query pre-processor* to prepare the query.
- *Sources chooser* agent, which identifies possible data sources.
- *Resource broker* where resources are registered.
- *Query planner* which generates an appropriate plan for processing the query.
- *Translation broker* to determine whether any *Translator* agents has to be used.
- *Translator* agent providing translation services.
- *Executor* generates one or more *Query workers* for a query.
- *Query workers* operate with *Data source agents* and *Computational module agents* to solve a query.
- *Data source* agents serve as wrappers for existing databases.
- *Computational module* agent provides a wrapper for a module that performs some specialist computation, such as statistical or connectionist analysis.

4.4.1.4 UMDL

UMDL [Weinstein *et al.*, 99] stands for University of Michigan Digital Libraries. The authors explain that digital libraries are just beginning to evolve. No one is certain what capabilities are needed, nor how they should be organised. It is therefore important to design digital libraries to be as open as possible, so that new collections and services can be easily added to the system. Furthermore, it is essential that libraries be able to scale to become quite large. This implies a decentralised architecture, where there are few if any shared resources and where as much decision making is done as locally as possible.

They state that agent-based systems are naturally decentralised, and can potentially be used to implement large-scale digital libraries.

In UMDL, the User Agent then requests the Planning Agent to help it put an agent team together to satisfy a query. The Planning Agent uses three other agents to help it do so. The Topics Agent provides a hierarchy of increasingly broad or narrow terms that are used by the Collection Agents to describe the contents of their collections. The Thesaurus Agent helps map from the query to these descriptions. The descriptions are stored and associated with agent locations by the Registry Agent. After comparing the query to the collection descriptions, the Planning Agent recommends one or more Collection agents to the User Agent, who then interacts directly with Collection agents to process the query.

Authors took an interest in the fact that agents pursue individual objectives and are capable of saying "no" to a request. For example, an agent might refuse a request if it is very busy, or if it is not very good at satisfying the particular kind of request. This behaviour would make sense in a system where agents acquire "reputations" based on their performance: Agents would want to avoid hurting their reputations by accepting responsibility for tasks that they expect not to be able to satisfy well.

In UMDL, the structure of messages and of conversations is implemented according to KQML (Knowledge Query and Manipulation Language). The authors explain that originally, they believed that they could establish a basic set of conversation protocols that would remain relatively stable even as the variety of content and services in the digital library became enormous. In practice, there was a tendency for new performatives to be added as needed for various situations.

They point out that KQML is ambiguous, and thus is not an adequate basis for co-ordinating the behaviour of agents created by developers who are not working closely together. They interpret these shortcomings of KQML as indicating that standardised conversation protocols are not in themselves an adequate basis for building large-scale agent systems. Other levels of commitment to common behaviour are also required. This is not, in retrospect, a surprising result.

4.4.1.5 InfoSleuth™

As far as I know, the InfoSleuth™ [Nodine *et al.*, 1999] (version1) started in 1995 and is still an ongoing project and a trademark of Microelectronics and Computer Technology Corporation. It is an agent-based system that can be configured to perform many different information management activities in a distributed environment. InfoSleuth™ agents provide a number of complex query services that require resolving ontology-based queries over dynamically changing, distributed, heterogeneous resources. These include distributed query processing, location-independent single-resource updates, event and information monitoring, statistical or inferential data analysis, and trend discovery in complex event streams. It has been used in numerous applications, including the Environmental Data Exchange Network and the Competitive Intelligence System.

InfoSleuth is an agent-based system that embodies a loosely coupled combination of technologies from information access, information integration, and information analysis disciplines. An agent system is a dynamic set of loosely inter-operating (though co-operating), active processes distributed across a network or internet. Each agent in an information-gathering system is a specialist in a particular task in the information enterprise. Agents perform focused tasks for collecting data to

create information at higher levels of abstraction or for collecting requests into generalisations of closely overlapping needs. Multiple agents interact and co-operate to solve complex information analysis tasks across multiple levels of abstraction. Typically an agent system uses some form of facilitation or market bidding to link up agents requesting services with agents providing services, and this capability is used to dynamically identify and compose goal-driven agent work groups.

The complete list of the agent roles in InfoSleuth (some of them have already been described in the section about other agent roles) is given by the authors:

- *User agents* act on behalf of users to formulate their requests into a form understandable by the agents themselves, and transform results into a form accessible to the user.
- *Resource agents* wrap and activate databases and other repositories of information. They consider any source of information a resource. They have implemented several different types of resource agents that can access different types of information. These include JDBC, text, flat files, and images. They also have service resource agents that start up offline data gathering and analysis services, such as web crawlers and document classifiers.
- *Broker agents* collectively maintain a knowledge base of the information the agents advertise about themselves. Brokers use this knowledge to match requested services with agents. Technically, the brokers collaborate to implement both syntactic and semantic matchmaking. When an agent comes on-line, it advertises itself to a broker and thus makes itself available for use. When an agent goes offline, the broker removes its advertisement from the knowledge base.
- *Ontology agents* collectively maintain a knowledge base of the different ontologies used for specifying requests, and return ontology information as requested.
- *Monitor agents* monitor the operation of the system.
- *Multi-resource query* agents process complex queries that span multiple heterogeneous resources, specified in terms of some domain ontology. They may or may not allow the query to include logically-derived concepts as well as functions over slots in the ontology.
- *Deviation detection agents* monitor streams of data for instances that are beyond some threshold, where the threshold may be fixed or may be learned over time. Deviations themselves form an event stream for other agents to subscribe to.
- *Subscription agents* monitor how a set of information (specified as an SQL query) changes over time.
- *Task planning and execution agents* plan how users' requests should be processed within the agent system, including how results should be cached. They may be specialised to particular domains, and support task plans tailored to their own domains.
- *Sentinel agents* monitor the information and event streams for complex events. A complex event is specified as a pattern of component events, which in turn may be other events such as changes in the information over time, triggers detected within individual resources, or deviations as detected by deviation detection agents.
- *Value mapping* agents used for some domains that require value mapping among equivalent representations of the same piece of information.

InfoSleuth is concentrating on the integration of external and heterogeneous sources of data. A network of external information sources contains data resources, at varying levels of abstraction, that provide partial evidence or facts for elements in the ontology. Resource agents wrap information sources, extract their content, mapping it to one or more domain ontologies, and monitoring their information.

As acknowledged by the multi-agent information systems community, ontologies appear to be a keystone and InfoSleuth does not depart from this rule. Users interact with the system by engaging in a session of ontology-based requests with a user agent. There is not one singleontology used in InfoSleuth, but rather a set of domain-specific ontologies, with one or more ontologies used for a given application. In InfoSleuth, each query is relies over some single, domain-specific ontology. The ontologies used in the InfoSleuth system are formulated using OKBC, and this allows queries over ontological concepts to be formulated and used by the system independently of the knowledge

base behind the ontology agent's OKBC server. The system then provides access to all of these resources using the common ontology as its query framework [Nodine *et al.*, 1999]

It is to be noticed that in InfoSleuth too the need for an ontology editor has been identified and addressed. A graphical ontology editor (Igor) is provided to help users build and modify InfoSleuth ontologies. The editor offers advanced features for constructing and visualising large ontologies.

4.4.1.6 Summarising remarks on the field

So far a large number of MAIS projects focused on the problem of dynamically integrating heterogeneous sources of information. It comes from the fact that it was one of the problems being addressed when multi-agent systems met information systems. As an example, the Information Manifold project [Kirk *et al.*,1995] is a system for browsing and querying multiple networked information sources. It demonstrates the viability of knowledge representation technology for retrieval and organisation of information from disparate (structured and unstructured) information sources. Such an organisation allows the user to pose high-level queries that use data from multiple information sources. The project was also interested in query processing algorithms used to combine information from multiple sources.

InfoMaster [Genesereth *et al.*, 1997] uses a global schema and Carnot [Collet *et al.*, 91] a global ontology (Cyc) to build mappings for wrappers of heterogeneous sources. As in RETSINA [Decker and Sycara, 1997], these systems rely on wrapper agents to provide an homogeneous view of the different sources while the integration is handled by middle agents planning query resolution, information integration and conflict resolution. Information Manifold [Levy *et al.*, 1995] and InfoSleuth [Nodine *et al.*, 1999] have multiple ontologies, but they do not handle mapping between them. SIMS [Arens *et al.*, 1996] uses Description Logics to handle multiple ontologies and translate queries when there is no loss. OBSERVER [Mena *et al.*, 1996] takes into account the inter-ontology relationships to tackle the loss of information when translating queries.

SAIRE [Odubiyi *et al.*, 1997] and UMDL [Weinstein *et al.*, 1999] manage distributed large scale libraries of digital documents to offer means to find relevant documents and manage indexing.

Closer to my interests are projects that focus on knowledge management inside organisations. CASMIR [Berney and Ferneley, 1999] and Ricochet [Bothorel and Thomas, 1999] focus on the gathering of information and adapting interaction to the user’s preferences, learning interest to build communities and collaborative filtering inside an organisation. KnowWeb [Dzbor *et al.*, 2000] relies on mobile agents to support dynamically changing networked environment and exploits a domain model to extract concepts describing a documents and use them to answer queries. RICA [Aguirre *et al.*, 2000] maintains a shared taxonomy in which nodes are attached to documents and uses it to push suggestions to interface agents according to user profiles. Finally, FRODO [Van Elst and Abecker, 2001] is dedicated to building and maintaining distributed organisational memories with an emphasis on the management of domain ontologies.

4.4.2 Common roles for agents in multi-agent information systems

It is a sign of their maturity that co-operative information systems are beginning to evolve a standard set of agent types [Singh and Huhns, 1999]. Therefore, in the following subparts I shall give a description of the most standard agent roles in multi-agent systems; the plan follows the review given in [Singh and Huhns, 1999] and originally in [Huhns, 1998].

4.4.2.1 User agent role

Concerning the multi-agent systems [Etzioni and Weld, 1995] explained that the details of the technology underlying the agents and the resources they accesses must be user-transparent. The agents enable users to state what information they require and determine by themselves where to find the information and how to retrieve it. The role of interfacing the user with the system is usually called the *user agent role*. Interfacing is not limited to interface display; conversational interfaces and profiling may also be part of the user agent role to enable individual and efficient information gathering. Profiling the user consists in applying user modelling techniques to gather information about likes and dislikes of the user that will be used to tailor the services and results provided by the multi-agent information system [Klusch, 1999c].

An important part of the work on agents for interfaces focuses on personal assistants *i.e.,* the emphasis is on the assistance relationship between the user and the agent (instead of the user-tool paradigm that is currently the one of software). The manner in which this is done ranges from providing help and high-level services, to relieve the user of tedious tasks such as scheduling meetings and filtering out unwanted email. These are direct descendants of the Softbot of [Etzioni and Weld, 1994].

In that relation user-agents, the user agents interact with the user receiving user's specifications and delivering results. They acquire and utilise user model to guide system co-ordination in support of the user's tasks. The main functions of an interface agent include [Sycara, 1999a]:
- Collecting relevant information directly or indirectly from the user to initiate a task;
- Presenting relevant information including results and explanations;
- Asking the user for possible additional needed information during problem solving;
- Asking for confirmation when necessary.

Interaction through an interface agent adequate for a task, hides the underlying distributed information gathering and problem solving complexity. This can be completed by the design principles given by [Etzioni and Weld, 1994] for the SoftBot:
- *Goal oriented*: a request to an interface agent indicates what the human wants; the agent is responsible for deciding how and when to satisfy the request.
- *Charitable*: a request is not a complete specification of the human's goal it is a clue to interpret.
- *Balanced*: the agent has to balance the cost of finding a piece of information on its own, against the nuisance of pestering the human with a question.
- *Integrated*: the agent provides a single, expressive, and uniform interface to a wide variety of services.

To do so, [Singh and Huhns, 1999] believes that user agents must contain mechanisms to select an appropriate ontology; propose a variety of interfaces to express a need, such as query forms, graphical query tools, menu-driven query builders, and query languages; propose a variety of interfaces to present results; maintain models of other agents; and, provide access to other information resources, such as data analysis tools, workflows, and concept learning tools. The maintenance of a model of the other agents, may be mitigated by the introduction of a middle-agent as we shall see.

In order to be truly helpful, any interface agent must learn about the user's habits and preferences. This is why interface agents may also be called *'Learning' agents*. Very much tied in with interface agents are the notions of trust and competence. Because these agents are directly interacting with humans, they must demonstrate good learning skills in order to build up competence and hence trust.

Any progress towards a flexible, more convenient human-agent interaction will help to increase the human users' acceptance of information agents for doing their everyday business [Klusch, 1999a]. Being the front-end of the system, the user agent acceptance determines the acceptance of the whole system acceptance.

A lot of the actual projects are focusing on helping the user in exploiting the world wide web and other services of the Internet. But these considerations are still valid in the context of other interconnected information and services networks such as corporate intranets, extranets, virtual enterprise networks.

Finally, the user agent calls the problem of the impact of loosely coupled software components architecture on the user interface. Because the user interface tries to abstract away from the normal functioning-level the distributed information system, it becomes more difficult to understand the scope of an action and its consequences. None of the technical details (sources location, size, etc.) involved in the fulfilment of a request is readily apparent to the user. A seemingly innocent query can generate huge amounts of undesired data from several sources [Nodine *et al.*, 1999]. This problem of hiding the complexity to the users causes an extra burden on mediators and resource agents to try to optimise and control the data flow (in the case of InfloSleuth [Nodine *et al.*, 1999], new agents have had to be introduced to monitor these problems). It also causes problems of explaining the results / failure to the user without using useless technical details.

Thus to summarise, the user agent role is essentially divided into two roles:
- *interfacing the user with the system* (services availability and results presentation) so that the user appears as an agent in the multi-agent system.
- *modelling the user* to adapt the system behaviour.

4.4.2.2 Resource agent role

Resource agents come in a variety of common types, depending on which resource they are representing, and provide a variety of capabilities. [Singh and Huhns, 1999]. One of the main roles of resource agents is to act as translators between the global transaction schema of the multi-agent system and the local interaction schema of the resource they are attached to. In particular, they have the ability to wrap heterogeneous legacy information systems (databases, in-house systems, etc.)

[Genesereth and Ketchpel, 1994] discuss the problem of integrating legacy software with agent systems, and suggest three possible solutions to the problem:
- *rewriting* the software which is a costly approach.
- use a separate piece of software called a *transducer* that acts as an interpreter between the agent communication language and the native protocol of the legacy system.
- the *wrapper* technique where the legacy program is augmented with code that enables it to communicate using the inter-agent language.

Wrapper agents denote agent implementing a transducer or wrapping technique. On the one hand they implement common communication protocols and on the other hand they have the expertise of the local access languages to translate back and forth. These local languages may be SQL, OQL, a knowledge-base language, operating system command shell, or other application-proprietary languages.

If the system is based on a global ontology, then providing mappings that relate each information resource to the shared semantic grounding requires, at most, *n* sets of mappings for *n* resource agents; however the global ontology has to be built and maintained to allow all these mappings. On the contrary, without a global ontology *n(n-1)* mappings are needed for direct peer-to-peer interactions among *n* resources; however more precise mapping may be achieved and no global ontology has to be built or maintained.

In addition the agent may be adding special information services. In particular, *data analysis agents* apply machine learning techniques to form logical concepts from data or use statistical techniques to perform data mining [Singh and Huhns, 1999]. The agents provide resource services at a higher level than the bare access to the information content; there are two uses for these services:

- *provide computation and additional* services such as co-relation, analysis, statistics, monitoring etc. for direct use by the users or for the system to derive indicators that it uses for its own adaptation or trigger action when a monitored event is detected.
- *align the information access-level* to the one of the communications in the system. For instance, if a set of Web pages provide interesting data and the multi-agent system uses semantic level message passing, then a web-mining service must be provided to extract the information and transform them into formal knowledge structures that can be exploited.

Designing wrappers for specific sources turns out to be extremely time consuming. [Ashish and Knoblock, 1997] explain it is impractical to construct wrappers for Web sources by hand for a number of reasons:

- The number of information sources of interest is very large, even within a particular domain.
- Newer sources of interest are added quite frequently on the Web.
- The format of existing sources often changes.

A mechanism of generator for translators is described in [Bergamaschi and Beneventano, 1999]: the agent is generated based on the description of the conversion that needs to take place for queries received and results returned. [Ashish and Knoblock, 1997] use LEX and YACC to generate a parser to simplify the task of obtaining information from the vast number of information sources that are available on the World Wide Web. They propose an approach for semi-automatically generating wrappers for structured internet sources. There are some very tricky parts in automating the wrapper generation and that is why so far wrapper generation is, in the best case, semi-automatic. Other approaches are based on training agents on samples to make them learn how to extract information. For example, [Muslea *et al.*, 1999] describes an approach to wrapper induction based on the idea of hierarchical information extraction where extraction rules are described as finite automata learned by the agent. This would allow users to develop and launch a population of agents, each monitoring an assigned source.

4.4.2.3 Middle agent role

Before they can co-operate, agents must be able to find each other. "The adoption of an appropriate co-ordination mechanism is pivotal in the design of multi-agent system architectures" [Hayden *et al.*, 1999]. This is why there is a lot of research going on in the area.

Middle agent mediate among services requesters and services providers for some mutually beneficial collaboration. The general principle is that providers must first register themselves with one or several middle-agent to advertise their capabilities; to do so, they send an appropriate message describing who they are, how they can be contacted and what kind of service they offer [Klusch, 1999b]. Using services descriptions and request descriptions, middle-agents can find the adequate matching for a collaboration to take place.

The very basic type of middle-agent is the Billboard, where agents contribute to and consult themselves the list of advertisements. This type of agent is excessively simple and barely deserves to be qualified as an agent role. Much more interesting and commonly used roles are the *matchmaker* and the *broker*.

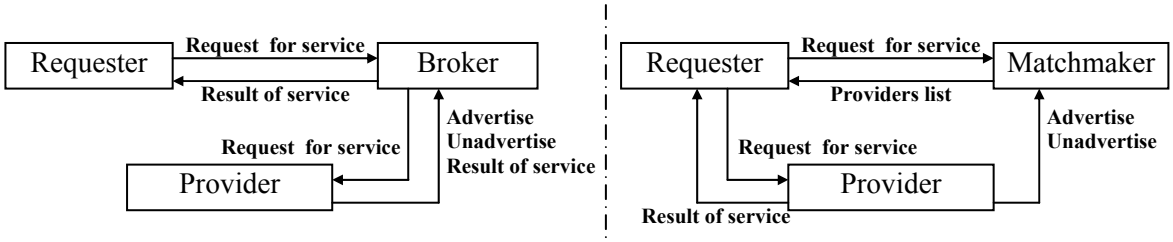


Figure 35 Service brokering and matchmaking in a CIS from [Klusch, 1999b]

As shown in Figure 35 [Klusch, 1999b] each request a matchmaker or broker receives will be matched with its actual set of advertisements. If the match is successful, a matchmaker agent returns a list of appropriate provider agents to the requester. In contrast a broker agent does not return a list of providers, but directly deals with the task of contacting the relevant providers, transmitting the service request to the service provider and communicating the results to the requester.

Middle-agents implement the two services of "yellow pages" and "white pages" directory for locating agents respectively with appropriate capabilities descriptions or identity descriptions. They manage the namespace and service space.

Service description requires a language and an ontology. The description matching process can be extremely complex, with all types of measures to try to match request as closely as possible with potential providers' profile.

An example of description language and different types of matchmaking process are provided in LARKS [Sycara *et al.*, 1999b]. LARKS defines the following frame slot types to describe advertisement or requests:
- *Context*: specification in the local domain of the agent.
- *Types*: optional definition of the data types used in the specification.
- *Input, output and input/output variable* declarations for the specification. In addition to the usual type declaration, there may have concept attachments.
- *In constraints and out constraints*: logical constraints on input/output variables that appear in the input/output declaration part. The constraints are restricted to be Horn clauses.
- *Concept descriptions*: optional description of the meaning of words used in the specification. The description relies on concepts in a given local domain ontology.

Thus, in any case, to achieve matchmaking, an agent must have a model of its own domain (an *agent ontology*) and a model of the other agents that can provide services (an *agent awareness model*).

These two models constitute the knowledge model of a middle agent and are used to determine how to process a service request [Papazoglou and Heuvel, 1999].

An anecdotal closing remark is that the designation of middle agents is still far from being universal, as an example, the definition of the Broker of InfoSleuth given in [Nodine *et al.*, 1999], corresponds in fact, to the definition of a Matchmaker presented here.

4.4.2.4 Ontology agent role

"Ontology agents collectively maintain a knowledge base of the different ontologies used for specifying requests, and return ontology information as requested." [Nodine *et al.*, 1999] Ontology agents are essential for interoperation since they provide a common context as a semantic grounding, which agents can then use to communicate. They provide access to multiple ontologies and manage the distributed evolution and growth of ontologies [Singh and Huhns, 1999]. They are a special kind of resource agents that could be called meta-resource agents. They provide downloads and updates of the ontologies for other agents, and maybe additional services such as terms and synonyms for the concepts, resolutions of queries on the hierarchy of concepts and relations, translation between ontologies, monitoring services for ontology change notification, etc. They provide, for instance, the user agents with the ontologies needed for query elicitation and the mediators or resources agents with the ontologies needed for query solving.

I made it clear that ontologies are a whole field of research, thus ontology agents inherit all its issues and results (methods, formalisms, platforms, etc.). Ontologies are a keystone of multi-agent systems and play an important role in the new generation of information systems. Therefore, it was likely that they would become a central component of multi-agent information systems.

When a solution buys ontology-based tools, it also buys it drawbacks. However, in the case of multi-agent systems some of the drawbacks of ontology, in particular consensus emergence and maintenance, may find a solution in the agent field. For instance, in FRODO [Van Elst and Abecker, 2001], there is a dedicated society of agent to build and maintain distributed domain ontologies and mapping between them. Thus the ontology-agent couple must be viewed as a mutually beneficial association and not the mere application of ontologies to multi-agent systems communication problems.

4.4.2.5 Executor agent role

There are agents in charge of the execution of a complex process, which might be implemented as knowledge-based systems, workflow systems, logic programming systems, planning systems, etc. [Singh and Huhns, 1999]. The authors explain they may be employed to supervise request fulfilment, to support scenario-based analyses, to assist workflows, etc.

For instance, in Infosleuth [Nodine *et al.*, 1999], task planning and execution agents plan how users' requests should be processed using the services proposed by agents in the system, and how results should be cached. These agents may be specialised in a domain, and support task plans tailored to this domain.

4.4.2.6 Mediator and facilitator agent role

Gio Wiederhold described in 1992 the concept of *mediator* as a software module that exploits encoded knowledge about some sets or subsets of data to create information for a higher layer of applications [Wiederhold, 1992]. This notion covers both the wrapper agent and the mediator agent. To avoid broadcasting messages, systems implement *federated facilitators* derived from this concept of mediators. These facilitators focus and target the communication and collaboration processes.

Mediators are specialised execution agents. They correspond to the executor role that determines which resources might have relevant information for a request they are in charge of fulfilling. They may have to decompose the request so that sub-requests can be handled by multiple agents. In that case, they also have to recombine the partial results obtained from multiple resources. [Singh and Huhns, 1999]

Mediators are sometime considered as middle agents since they mediate between querying agents and resource agents. But their real role is more to provide a service such as "distributed query solving" rather than matchmaking agents. For the task of matching an agent to a required service (e.g. "which agent can provide me an ontology of XYZ"), mediators themselves will call upon middle agents. Therefore, I would rather classify mediators as specialised execution agents.

4.4.2.7 Other less common specific roles

The application scenarios may lead designers to introduce specific roles. The 'loosely coupled software components' aspect of multi-agent systems is prone to such extensions.

In InfoSleuth [Nodine *et al.*, 1999] and in NZDIS [Purvis *et al.*, 2000], for instance, authors had to add other roles to match the different requirements of the application domains they were working on.

I shall pick out two roles:
- *Monitor agents* that provide regulation services (system administration watchdogs such as deviation detector) of event notification services (sentinels watching workflows, data monitors, etc. providing event subscription services).
- *Computation services agents*: that are special types of executors providing value mapping, calculation services, etc. sometime wrapping other legacy software (e.g. MathLab).

The sentinel and subscription agents are characteristic roles needed when the system is to show some proactive behaviour monitoring, reacting to events and pushing new information.

Another types of role comes from the field of collaborative information filtering that essentially automates the process of "word of mouth" in a given user community [Klusch, 1999c]. The main purpose and application of additional roles introduced in such systems is to enable agents to anticipate the individual usage and needs, by analysing the statistic uses and needs of users having a similar profile. Such a functionality in a multi-agent system introduces the need for new user agent roles like: profile comparison and clustering (use suggestion), profile matching (user matching), etc.

4.5 Conclusion of this survey of multi-agent systems

In this last part of the guided tour of relevant literature, I have tried to show research in multi-agent system proposes both a very well suited software architecture for distributed information systems and an integration platform for the different approaches I am interested in (knowledge-based systems, user and organisation modelling, information system, ontology-based systems). Indeed, distributed artificial intelligence naturally involves multiple autonomous agents where knowledge is distributed and builds agents co-operation relying on message-based communication [Bond and Gasser, 1988].

I started with a review of the core notions of the field and elicited the definitions that will be used in the rest of the document, concerning agents and multi-agent approaches to information systems.

I reviewed a number of methodological contributions to derive coherent implementation specifications of a multi-agent-system, (architecture, interactions, agent roles and behaviours) from the overall requirements of the end-users. The whole design chain and its interactions may be seen like this:

abilities & behaviours ↔ **(roles ↔ interactions)** ↔ groups ↔ global social functions

I chose to follow the organisational dimension to engineer an information system and I shall use its core notions to structure my design approach. The notion of *role* being pivotal, I reviewed the most common roles found in multi-agent information systems and to allow their interactions, I briefly surveyed possible languages.

It is clear that the integration power of multi-agent system does address the needed effort for regrouping and reconciling the different expertise fields and associated techniques; it just allows their coupling through semantic-level message passing interactions. The development of complete complex multi-agent information system requires strong expertise from several related research areas, such as programming languages, artificial intelligence, distributed systems, information retrieval, logics, cognitive and social sciences, knowledge engineering, computer supported collaborative work, linguistics, human-computer interaction, etc. It is not to be forgotten that each research area that you may call for will not only come with its toolbox of solutions, but also with its attached lot of problems. One must be careful in not compiling a non viable set of elements of solution, in a limited project; the time-consuming management of multi-disciplinary teams, must not be underestimated.

Besides this point, this survey has shown a number of results and initiatives that I found convincing proof of concepts, showing that agents are a serious candidate to provide a new design and implementation paradigm that may be used to address complex distributed problems and, among others, information and knowledge management problems in distributed contexts.

5 Research context and positioning

Always design a thing by considering it in its next larger context;
a chair in a room, a room in a house, a house in
an environment, an environment in a city plan.
— Eliel Saarinen

The highly dynamic information society and its globalisation led organisations to shorten their response time especially by developing their internal information systems. Intranets became organisational nervous systems and the base of corporate memories. They are often based on Web technologies and can therefore benefit from progresses made for the Semantic Web to improve the Web exploitability through semantic annotations of its resources. In parallel, multi-agent systems propose a distributed artificial intelligence architecture that can be deployed over scattered information landscapes, such as corporate semantic webs, to support their management.

I carried out my Ph.D. in the framework of a European project called CoMMA that stands for Corporate Memory Management through Agents. This project was at the crossroad of several domains and mainly: knowledge engineering, multi-agent systems, machine learning, information retrieval, web technologies. As I shall explain in this chapter, the objective of CoMMA was to design, set up and test a multi-agent information system managing a corporate semantic web and its users in the context of two application scenarios: (1) assistance in providing information to ease the integration of a new employee and (2) support to information management activities in technology monitoring processes.

In the first part, I shall identify the requirements, presenting two application scenarios and deriving a set of need functionalities shared by both scenarios. The second part will describe and justify the implementation choices. Finally, the third part will position CoMMA in the state of the art and give a global overview of the solution from which I shall be focus on my contributions.

5.1 Requirements

5.1.1 Application scenarios

Companies possess a knowledge that is really large and complex, therefore, it becomes fundamental to exploit it, providing employees with effective and efficient means for communicating their knowledge and ideas for the benefit of the whole organisation. This involves the processes of representing, capturing, accumulating and transferring distributed organisational knowledge in working environment.

The CoMMA project (IST-1999-12217) was funded by the European Community and aimed at implementing a corporate memory management framework. Many trials for an overall solution of a KM system have failed due to the complexity of the problem. To reduce this complexity we limited the scope of the project to two scenarios:

- The *integration of new employees* in the company, where the problem is: how can a corporate memory speed-up the process of integration of a new employee by bringing the needed information to the newcomer at the right time and the right order ?
- The *support of technology monitoring*, where the problem is: how can a corporate memory assist the capture of information acquired by the technology monitoring departments and put it at the users' disposal or even push it to the right people ?

In this part, I describe these application scenarios, underlying the aspects that CoMMA aimed at supporting.

5.1.1.1 Knowledge management to improve the integration of a new employee

This scenario aims at analysing the different solutions that can be provided to improve the newcomer integration in the organisation. For the organisation, it is important to make newcomers rapidly operational by providing them with the information they need about their environment, the people and the resources they have access to. Integration is usually supported by the personnel department or/and by more experimented and skilled employees, usually colleagues who play more or less officially the role of *tutors*.

Our objective was not to build an expert system that would play the tutor role; so far the best way to be introduced to a human organisation is through some of its humans members. The stake of a knowledge-based system here, is to provide the knowledge needed and to guide the newcomers in their discovery of the organisation. Models are needed to describe the organisational infrastructure and the profiles of the actors; these models can guide the system and ensure that newcomers get the right information when they need it.

The different actors of the scenario use different functionalities of the system. The tutor and human resource manager use document annotations to populate the memory and enable the system to prompt relevant resources. These annotations can reference the models to anchor the memory in its environment for instance an annotation can state that:

> "a memo available at the URL 'http://intranet/DR/car_par.doc' has for title 'on obtaining a car park badge'; it was issued by the department 'infrastructure and maintenance'; it addresses the subjects of 'car' and 'car park'; and it is especially targeted to 'newcomers' "

The new employees may use such a system at several stages of their insertion process: when they just arrived in the organisation, it is interesting to push interesting information to them; as they grow confident and learn to use the system, they will use pull functionality to extract specific information from the memory.

Profiles are also needed to store facts about the different stakeholders and enable the system to tune its behaviour, for instance the more the newcomers use the system, the more the system will be able to know what information to push to them.

In this scenario, there are two aspects directly relevant to a system supporting knowledge management:
- The *role of a memory*: it is used to avoid repeating the same pieces of advice, the same explanations, each time a newcomer arrives and to capture and index once and for all the relevant documents support the integration.
- The *role of a nervous system*: it is used to propagate information picking it up where it is detected (tutor, human resources, etc.) and transmitting it to where it is useful (newcomers).

5.1.1.2 Knowledge management to support technology monitoring and survey

Organisations are living organisms evolving in changing environments (markets, bigger organisations, culture, country etc.). Their survival requires them to sense and act upon their environment. Technology monitoring and survey are about detecting, identifying and interpreting technological and scientific movements, to diffuse innovative ideas among employees and react to or anticipate evolutions. In this scenario, knowledge management can aim at assisting:
- identification and survey elicitation about monitoring results;
- storage and diffusion of technological knowledge through the whole company;
- access to the relevant information for further reference.

Any employee from the company can be involved in technology monitoring activities (manager, technician, engineer, researcher, etc.). They can supply elements of survey that take the form of annotations about information resources e.g. pages they have found on the Web, reports they wrote and put on the intranet, etc. These annotators contribute to the organisational memory.

In case users need some specific information, they need intelligent mechanisms to retrieve it from the memory. This scenario also requires a push approach, in which information pieces are automatically suggested to users according to their profiles; the profile describe users and the system uses these models to determine which documents will be of interest to each one of them. Proactive diffusion is extremely important here since it enables fast diffusion of detected news.

In this scenario again, there are two aspects directly relevant to a system supporting knowledge management:
- The *role of a memory*: it plays its usual role of persistent repository of acquired knowledge where intelligent indexing allows relevant documents to be retrieved when needed.
- The *role of a nervous system*: it is a medium between people, propagating information, capturing it where it is detected (technology monitors, occasional annotator) and pushing it where it is useful (strategy annalist, managers, engineer, etc.) through the filter of their profile.

5.1.2 The set of needed functionalities common to both scenarios

The two offices of memory are
collection and distribution.
— Samuel Johnson

In both scenarios, the envisioned system is used as an information portal for the everyday work to receive and retrieve stored information. In both cases, a supporting system has two facets:
- *Memory* of the existence and relevance of information resources; the memory can be considered as a distributed set of documents annotated to be intelligently indexed.
- *Nervous system* of the organisation linking persons and organisations parts (department, services, etc.); the management systems must deal with the distributed nature of the problem

There are recurrent sub-scenarios common two both application scenarios, that call for common functionalities in a supporting system:

Contribution to the memory (add functionality): information resource annotation using ontological primitives. Interfaces guide the users in describing a resource navigating inside the corporate ontology to select the consensual concepts that will be used for indexing and diffusing the new resource. Complexity of the memory, its materialisation and management are hidden. In the new employee scenario, tutors or human resources people annotate documents to make them accessible to the newcomers. This can concern documents related to the general enterprise life or to the newcomer specific job. In the technology monitoring scenario, dedicated people (information officer, technology monitors, librarians, etc.) can point out information resources through annotations, using the same process.

Active remembering and diffusion (push functionality): pushing information to users according to their profile. The user profiles is a description of their interests, activity, etc. that can be updated both by the system (learning from past interactions) and by the users (customising). The profile is used to evaluate the interest that a new resource conceals for a user thanks to a comparison to the newly submitted annotation of the resource. Suggestion must be presented in a non intrusive manner and feedback can be given by the users to adjust the relevance criteria used and learned by the system. In the new employee case, the push mode can offer a flexible mechanism to provide interesting information to newcomers as soon as he is identified by the system. As for the technology-monitoring scenario, it needs an efficient proactive mechanism to diffuse the documents at the enterprise level taking into account fields of interest.

Intelligent access to the memory (pull functionality): querying annotations and retrieving documents. The users must be able to search and retrieve resources from the memory. Queries are built using the ontological primitives through interfaces that guide the users in the elicitation an refinement process. Results are sorted by the system comparing them to the user profiles and the ranking is evaluated and realigned using the user feedback to maintain the quality of relevance. In the new employee scenario, a newcomer can address specific requests to the corporate memory to get very precise information about many subjects (Where to get lunch tickets? How to organise professional travels? Who is who? etc.). In the technology monitoring scenario, the importance of being able to retrieve information on past projects, survey of competitors, technical reports, etc. is obvious.

Coherence in indexing and communication: semantic grounding ensured by the deployment and use of a shared ontology in the corporation. The normalised meaning of ontological primitives enable a distributed and yet coherent indexing of knowledge and intelligent mechanisms to access, query and exploit the memory.

Awareness of the corporate world: the system must be aware of its environment *i.e.,* the organisation and its members to manage the annotations and, their references and their flow. To structure the knowledge which will be maintained by such a tool, different kinds of models are useful. User models can support customisation of the system functionalities to address specific needs of specific "types" of users. A corporate model captures the groups and their relations to handle differences existing between various activities of the enterprise. Both scenarios exploit functionality that will require the availability and management or organisational and user models. The user profile also positions the users in the corporate model giving the system the complete view of the organisational structure.

Many other functionalities can be envisioned, but these are the ones that were explored in CoMMA.

5.2 Implementation choices

In both scenarios, the actors (e.g. tutor/tutee, technical area referents, etc.), the information (e.g. newcomer route card, technology trend analysis card, etc.) and the actions performed on the memory (e.g. adding an annotation, pushing/pulling information, etc.) are distributed and heterogeneous by nature. The conceptual and technical choices of CoMMA are mainly motivated by these three observations.

5.2.1 Organisational memory as an heterogeneous and distributed information landscape

Knowledge is power
— Francis Bacon

If knowledge is power, then the distribution of knowledge is a form of distribution of power, in other words, a separation of powers. Inside and between organisations, knowledge is naturally distributed between artefacts (codified in information resources such as books, software, etc.) and humans. There are multiple reasons that make it difficult to centralise all this knowledge in one system:

- *Practical feasibility*: formalisation and coding are complex, time-consuming, sometimes to the extent of not being realistic.
- *Knowledge is a valuable asset*: people and organisation may see no reason for giving it freely and, on the contrary, the existence of competition will be a major brake.
- *Heterogeneity of sources*: knowledge workers and the knowledge they maintain are scattered all around the organisation and the tools, the format and processes they use are "diverse and varied".
- *Low return on investment*: the cost of building centralised repositories and maintenance systems may overcome the benefit of centralisation.

In our case, the knowledge handled is not valuable for the actor themselves and therefore, benevolence is plausible; however the other brakes remain and, all things considered, the separation of knowledge pieces may be a salutary policy. Therefore, I shall consider the corporate memory as an heterogeneous and distributed information landscape.

The corporate memories are now facing the same problem of information retrieval and information overload than the Web. To quote John Naisbitt, “we are drowning in information, but starved of knowledge”. The initiative of a semantic Web [Berners-Lee *et al.*, 2001] is a promising approach where the semantics of documents is made explicit through metadata and annotations to guide later exploitation. Ontobroker [Decker *et al.*, 1999], Shoe [Heflin *et al.*, 1999], WebKB [Martin and Eklund, 2000] and OSIRIX [Rabarijaona *et al.*, 2000] are examples of this technique, relying on annotation based on ontologies.

XML promises a durable storage and exchange format independent of the software platforms and applications and their own format. The same source may be accessed and transformed to propose different views to different agents (human or software). XML being likely to become an industry standard for exchanging data, we use it to build and structure the corporate memory.

We are especially interested in RDF(S) and the associated XML syntax. It allows us to semantically annotate the resources of the memory. The corporate memory is then studied as a "corporate semantic Web". A corporate semantic web is a corporate memory distributed over an intranet and relying on the semantic Web framework for storage and exchanges of annotations; the intranet resources are semantically annotated using the conceptual vocabulary of an ontology designed for the organisation and the application scenarios.

A legacy application is a program or a group of programs in which an organisation has invested time and money and usually it cannot be changed or removed without considerable impact on the activity or the workflow. Just as an important feature of new software systems is the ability to integrate legacy systems, an important feature of a corporate memory management framework is the ability to integrate the legacy archives, especially the existing working documents. Since RDF allows for external annotations, existing documents of the corporate memory may be kept intact (word processor document, spreadsheet, image, etc.) and annotated externally.

The memory is composed of heterogeneous changing documents, we structure them using semantic annotations expressed with primitives provided by a shared ontology. RDF and RDFS provide the framework to write the annotations and formalise the ontology in a schema.

5.2.2 Stakeholders of the memory as an heterogeneous and distributed population

The population of the stakeholders of our scenarios is, by nature, heterogeneous and distributed in the corporation. The system is in charge of interfacing users with the content of the memory. Adaptation and customisation are a keystone here and the system relies on machine learning techniques in order to make agents adaptive to users and context. This goes from basic customisation to user's habits and learning of preferences, up to push technologies based on interest groups. I shall not detail this aspect of the work since it was achieved by our partners of the LIRMM [Kiss and Quinqueton, 2001].

The description of the different user groups, profiles and roles involved in the two scenarios, uses the primitives of the ontology to make explicit, share and exploit a model of the organisational environment and user population.

5.2.3 Management of the memory as an heterogeneous and distributed set of tasks

The tasks, as a whole, to be performed on the corporate memory are, by nature, distributed and heterogeneous. The corporate memory is distributed and heterogeneous. The users population is distributed and heterogeneous. Therefore, it seems interesting that the interface between these two worlds be itself heterogeneous and distributed.

Full centralisation just as full distribution is not realistic. There will always be different applications running at different geographical points and using different data. The users will always want to be able to communicate and exchange data between these applications while remaining free to manage their own data as they want. Thus flexible and distributed architectures are needed to assist interoperation, logical integration and virtual centralisation and the agent paradigm appears very well suited for the deployment of a software architecture above the distributed information landscape of the corporate memory:

- On the one hand, individual agents locally adapt to users and resources they are dedicated to.
- On the other hand, thanks to co-operating software agents distributed over the network, the whole system can capitalise an integrated and global view of the corporate memory.

When multi-agent systems are intended to support and/or control some real-world organisations, an organisation-based design reduces the conceptual distance between the software system and the real-world system it has to support. Consequently, this simplifies the development of the system. [Zambonelli *et al.*, 2000]

Agents individuality and autonomy enable them to locally adapt to local resources and specific users. Multi-agent systems are loosely coupled by semantic-level message passing enabling co-operation for a global capitalisation. Communication relies on a shared semantic of the primitives used in the messages and captured by an ontology.

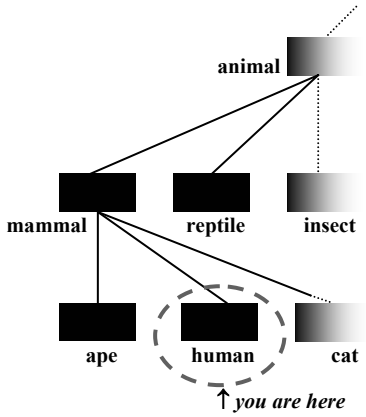
The main objective is to help users retrieve information relevant for them: details of the technology underlying the agent and the resources the agent accesses are 'abstracted' away - that is, they are user-transparent. The agents enable the users to state what information they require or proactively try to guess it. Then, the agents determine where to find the information and how to retrieve it.

5.2.4 Ontology: the cornerstone providing a semantic grounding

The ontology provides shared conceptual vocabulary that enables efficient and non-ambiguous communication (e.g. exchange between a technology monitor and an engineer interested in a piece of information). It also makes explicit, in a reference document, the organisational jargon enabling people to understand the 'company language' (e.g. a newcomer can find in the ontology the meaning of some internal concepts - what are the meanings and the difference between a 'project' and an 'action' at INRIA). Thus the ontology is not only a seminal conceptual tool, it is also an explicit element of the memory which captures some views, opinions, definitions and characteristics of the organisation. It provides a lexicon of the organisational world (company policies, organisational structures and processes vocabulary).

The ontology is a tool for conceptualisation, a semantic grounding for models, profiles, annotations and communication messages, as well as a full component of the memory highly relevant in itself for the stakeholders of the application scenarios.

5.3 Positioning and envisioned system



5.3.1 Related works and positioning in the state of the art

In this part I shall use the definitions and reviews provided in the previous chapters to position and describe the characteristics of CoMMA.

The envisioned memory is an hybrid memory both knowledge-based (ontology-based annotations of the documents, the people, the organisation) and document-based (annotated information resources). It is a resource-based memory, *resource* being used in the sense of the W3C *i.e.,* everything that can be referenced through an URI (online resources through their URL, people, groups, etc.). URIs can reference resources distributed anywhere, therefore, the actual memory is both internal and external; thus the memory of CoMMA is at the cross road of document-based memories, a knowledge-based memories and a distributed memories. Documents are heterogeneous, but annotations are homogenous and based on a shared schema, therefore, the issues of information integration is not addressed inside CoMMA, however additional wrapping of legacy information systems may require this. Knowledge-level indexing structure allows us to use several indexing system, manual, wrapping, mining, etc.

The multi-agent system paradigm adopted is closer to closed distributed problem solving systems in which the component agents are explicitly co-designed to co-operatively achieve a given goal than to general open systems. It corresponds to my definition of organisational multi-agent information system. The organisational analysis is used as a design tool with the restriction that in normal run time, sub-societies are not modified except for the addition or disappearance of instances of agents of known agent types.

CoMMA agents are coarse-grained agents from the symbolic branch of artificial intelligence. Internal architecture of agents makes use of knowledge-based architectures, machine learning classifiers, and concurrent finite state automata.

The CoMMA system can be seen as a distributed expert-systems society where the expertise is not the domain or the design rationale of the core activity of the organisation, but the document management in this organisation.

The agents of a corporate memory system form a distributed expert-system where the core expertise is the corporate information management.

Systems, to interact at semantic level passing, must agree on and then share a common definition of the concepts involved in their mutual communication; agents do not depart from this rule. Therefore, an ontology (O'CoMMA) of the domain and application has been built and provided the semantic for communication and annotation. The work on O'CoMMA and an the user and organisation models corresponds to what some agent-oriented methodologies call the *domain model*. We noticed that few of the methodologies seemed to be aware of the real work and design models required on that part. In CoMMA,it was at the core of the approach.

We saw that so far a large number of multi-agent information systems focused on the problem of dynamically integrating heterogeneous sources of information: SIMS [Arens *et al.*, 1996], TSIMMIS [Molina *et al.*, 1997], InfoMaster [Genessereth *et al.*, 1997], RETSINA [Decker and Sycara, 1997] and InfoSleuth [Nodine *et al.*, 1999]. In CoMMA we did not stress the heterogeneous sources reconciliation aspect: documents may be heterogeneous, but annotations are represented in RDF and based on a shared ontology formalised in RDFS. The ontology is not used to translate from local schemata to a global one in order to integrate information from different sources. It is rather used to describe the different information resources in order to enable their management. Information integration was not the primary concern although it may be used to enable to wrap sources useful for models e.g. a database of users and their password, a directory of phone numbers, etc.

The second type of systems I mentioned was agent-based digital libraries as Zuno digital library project [Ferguson and Karakoulas, 1996], SAIRE [Odubiyi *et al.*, 1997], UMDL [Weinstein *et al.*, 1999]. In CoMMA we do not archive or mine the documents themselves. Agents are used for wrapping annotation repositories indexing the document resources. We do share the concern for indexing and retrieval, but we do not manage the digital documents and we create models of the organisation and its members to tune the system to a specific corporate memory rather that a broad open library.

The third set of systems is specialising in making easy the gathering of information in an organisation: CASMIR [Berney and Ferneley, 1999] and Ricochet [Bothorel and Thomas, 1999]. CoMMA does not implement collaborative filtering, however it does try to foster communities of interest through the diffusion of annotations guided by the user profiles. It also aims at modelling the organisation to anchor the memory in its environment, the models being referred to in annotations.

KnowWeb [Dzbor *et al.*, 2000] implements mobile agents to address the partial connectivity of the users to the memory, an aspect which is completely overlooked here. The system also tries to extract concepts from the documents, whereas fully automatic mining is not an issue addressed in CoMMA since the annotations can be extremely complex and must be very precise.

RICA [Aguirre *et al.*, 2000] maintains a shared taxonomy in which nodes are attached to documents, in our case, the engineering of the ontology was done outside the system. Moreover, it is not a taxonomic indexing of documents; it is built to provide the conceptual vocabulary to express complex annotation enabling multiple axes of querying. However we do share with the RICA project the idea of pushing suggestions to interface agents according to the user profiles.

Finally, FRODO [Van Elst and Abecker, 2001] emphasises the management of domain ontologies and the building of gateways between different communities and their ontology. We have only one ontology shared and deployed in the whole system.

CoMMA does not address the corporate memory lifecycle in its entire complexity. The system focuses on the pull and push functionalities together with the archiving of semantic annotation of information resources in order to manage a corporate semantic web.

The CoMMA system helps the users in performing three main tasks: insertion of RDF annotations of new documents, search of existing documents, and autonomous document delivery in a push fashion to provide them with information about new interesting documents. The multi-agent paradigm was used for enhancing scaling, flexibility and extensibility of the system and to adapt the system interface to the users. Agents are not only used for the retrieval of information, but also for the insertion of new information in the corporate memory.

The design focused on engineering an architecture of co-operating agents, being able to adapt to the user, to the context, and supporting information distribution. The duality of the word 'distribution' reveals two important problems we wanted to address:

- *distribution means dispersion*, that is the spatial property of being scattered about, over an area or a volume; the problem here is to handle the naturally distributed data, information or knowledge of the organisation.
- *distribution also means the act of spreading*; the problem then, is to make the relevant pieces of information go to the concerned agent (artificial or human).

It is with both purposes in mind that we designed the CoMMA architecture as presented in the following section.

5.3.2 Overall description the CoMMA solution

Different research communities offer (partial) solutions for supporting KM. The integration of results from these different research fields provides more complete solutions. This was a motivation for CoMMA to implement and test *a corporate memory management framework integrating several emerging technologies: agent technology, knowledge modelling, XML technology, information retrieval and machine learning techniques*. Integration of these technologies in one system was already a challenge yet another was the definition of the methodology supporting the whole design process. The overall picture of CoMMA is given in Figure 36 and I shall comment it in details to explain both the processes and the architecture.

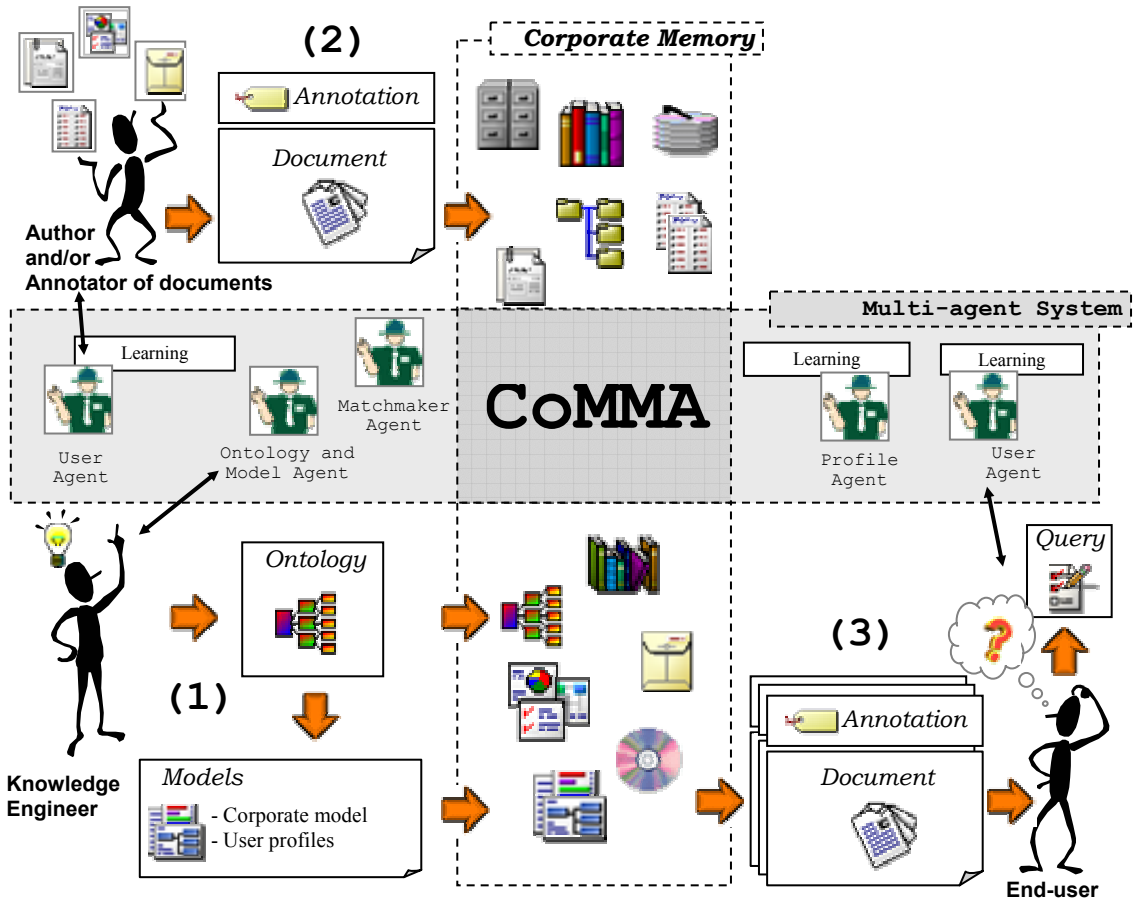


Figure 36 Overview of the approach

The knowledge management framework we envisioned was to be based on models that would support the information indexing and retrieval mechanisms. Thus the stage (1) in Figure 36 is the *engineering of an ontology and the description of the models* (corporate model and user profiles) that are exploited by the system. These models and the ontology are elements of the memory not only because they enable us to write the annotations that index the memory resources, but also because they are documents in themselves that can be consulted by users just as any other document. Some agents are in charge of managing the ontology, the corporate models and the user profiles. This stage is mainly carried out by one or more knowledge engineers working together with the knowledge workers/stakeholders of the application scenario. The results are *an ontology encoded in RDFS and models and profiles encoded in RDF as annotations about the users and the organisation*.

In stage (2) in Figure 36, the memory is populated *i.e., information resources are annotated*. These annotations describe the content, the interest, the type, the usefulness, etc. of the annotated documents. *Annotations are meta-information expressed in RDF format*. The concepts and relations used to express these annotations are defined in the ontology; the descriptions given may reference the models to describe the author, the target, etc. of a resource. Through annotations the corporate web is indexed and can be searched and explored; external resources can also be annotated through their URL which is most appropriate to the technology monitoring scenario. In the virtual landscape of information, the memory of an organisation effectively stretches beyond the real organisational frontiers. *Some agents are in charge of assisting the annotation and archiving process, interfacing the users with the indexing facet of the memory*.

In stage (3) in Figure 36, the remembrance occurs *i.e., the system brings documents to a user in reply to a query*. The query may have been explicitly submitted by the user or could have been proactively derived from the user's profile and the answer was pushed in a non intrusive manner. In both cases the user is presented with a list of resources. The project exploits *machine learning techniques in order to provide the agents with adaptive capabilities to their users and the context*. The project focused particularly on the improvement of search result relevancy, learning how to sort the results, taking into account the feedback given by the user.

Machine Learning methods aim at generalising sets of examples to produce general knowledge to be applied to new situations. The examples on which these learning capabilities rely are chronicles made of events representing the successive actions of the user, sampled at a level depending on the goal of learning (actions on page, change of page, change of site, change of user's goal). Within the CoMMA project, three complementary ways of using Learning Techniques could have been considered: (a) Learning on the fly: for a quick adaptation to the user during a session (b) Remote learning: to enhance the behaviour of the system between sessions (c) Lazy learning *i.e.,* case-based reasoning on a base of indexed cases: to take into account very specific cases or aspects. The CoMMA prototype implements learning on the fly using the feedback of the user.

The ontology, the models, the distributed dimension of the memory and its management systems are complex and this complexity must be hidden to the end-user. This places a heavy constraint on the whole system and more particularly on the user interface on which depends the whole system usability. The interface is *only the visible* tip of the system, yet it is *the only visible tip* and thus if it fails to please the user, the whole solution collapses no matter how efficient, intelligent, powerful the rest of the system is. Ergonomic studies are needed here.

The two main dimensions of the solution are the memory and the management software.

The management software is based on *a multi-agent architecture of several co-operating agents, having adaptive capabilities to the user and to his/her context, and supporting retrieval of the relevant information in the organisational memory.* These agents can (a) communicate with the others to delegate tasks, (b) make elementary reasoning and decisions, and help to choose between several documents. The MAS has the main responsibility of the system: it manages the corporate memory, it adapts the user interfaces and it interfaces the user and the memory. The realisation of the MAS was simplified by using a pre-existing software framework for the development of agent applications called JADE [Bellifemine *et al.*, 2001]. JADE made the development easy and compliant with the FIPA specifications [FIPA].

The memory materialisation relies on the Resource Description Framework (RDF) that uses a simple data model and an XML syntax for representing properties of Web resources (documents, images, etc.) and their relationships. RDF enables us to describe the content of documents through semantic annotations and use them to search for information. The annotations are based upon an ontology that is described and shared thanks to RDF Schema (RDFS). The idea is that (a) a community specifies concepts and their relationships in ontologies, (b) documents of the community are annotated using these ontologies, (c) annotations are used to search the memory and navigate into it.

Keyword-based search engines work at the term level and one of the reasons for introducing ontologies in CoMMA is that they are a means to enable software to reason at the semantic level. To manipulate the ontology, the annotations, and infer from them, my research group developed *CORESE* [Corby *et al.*, 2000], *a prototype of semantic search engine* enabling inferences on RDF annotations and information retrieval from them. CORESE combines the advantages of using (a) the RDF(S) framework for expressing and exchanging metadata, and (b) the query and inference mechanisms available for Conceptual Graph (CG) formalism [Sowa, 1984] for searching the memory. CORESE is a light-weight API in Java therefore, it is perfectly targeted at applications such as intelligent agents; it was used to provide a standard component for software agents in charge of distributed knowledge management.

In this part, I presented the overall context of my work and the system CoMMA as I see it, stressing the aspects relevant to this document. It should provide the reader with enough information on the context to understand the detailed presentation of my work that will be the subject of the following parts. As you most probably understood, CoMMA was a large project involving six institutes and tens of persons. I shall give pointers to documents describing the work of my partners as often as possible, but I shall not enter into the details of their work here since the following section will focus on my contribution to the project in the context of my Ph.D.

Part - II

Ontology & annotated corporate memory

Knowledge is true opinion.
— Plato

6 Corporate semantic web

This chapter is a short introduction to the vision of a corporate memory as a corporate semantic web providing an annotated world for information agents.

I present the motivation for introducing annotations to guide agents working on the memory. Then, I detail the components and the structure of this memory, underlining the fact it is based on a model providing some awareness of the environment to the system. In a third part, I give an overview of CORESE, the semantic search engine developed by my colleagues and that was used to build agents. Finally, I stress the seminal role played by the ontology in this corporate semantic web, thus justifying the work that will be presented in the following chapters.

6.1 An annotated world for agents

In their article about "Agents in Annotated Worlds" [Doyle and Hayes-Roth, 1998] explained that software agents must have the ability to acquire useful semantic information from the context of the world they evolve in, "knowledge can literally be embedded in the world as annotations attached to objects, entities and locations". They introduce the notion of "annotated environments containing explanations of the purpose and uses of spaces and activities that allow agents to quickly become intelligent actors in those spaces". Although the authors choose for their application domain the field of believable agents inhabiting and guiding children in virtual worlds, their remark is transposable to information agents in complex information worlds. This leads us to say that annotated information worlds are, in the actual state of the art, a quick way to make information agents smarter. If the corporate memory becomes an annotated world, agents can use the semantics of the annotations and through inferences help the users exploit the corporate memory.

In order to annotate the information resources of the corporate memory, we use the RDF(S) formalism based on XML technology and described previously. Each agent is individual, autonomous and situated. Its individuality and autonomy make it tuneable to the local resources or tasks it is dedicated to. Its situation and accesses to resources position it in the space of the corporate intranet and give its point of view of the memory as illustrated in Figure 37.

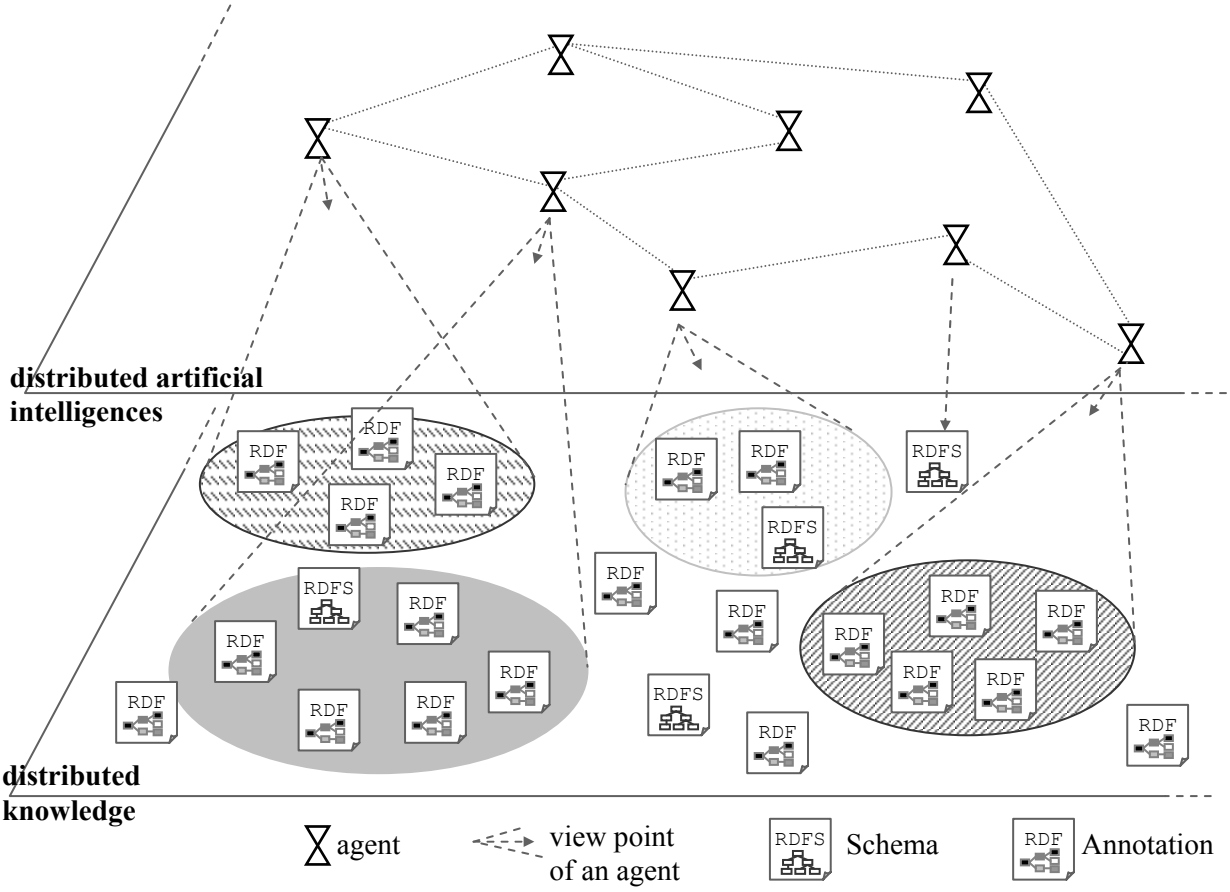


Figure 37 Agents situated in an annotated world

The availability of a shared ontology and of annotations based on that ontology, makes the information world 'comprehensible' to agents. They can then perform meaningful operations on behalf of users.

6.2 A model-based memory

As I explained in the first part, until now, enterprise modelling has been mainly used as a tool for enterprise engineering. But the new trends and the shift in the market rules led enterprises to become aware of the value of their memory and of the fact that enterprise model has a role to play in knowledge management too. [Rolstadås, 2000] notices that enterprise models may well constitute a theoretical basis for the information system in an enterprise, and are regarded by many as a substantial opportunity to improve global competitiveness of industry.

In CoMMA, our goal was not to evaluate the model or optimise it to support enterprise evolution. Our use of this explicit partial representation of reality is to support corporate memory activities involved in the new employee scenario and the technology monitoring scenario. The model enables the system to get insight in the organisational context and environment and to intelligently exploit the aspects described in this model for the interaction between agents and overall between agents and users. This participates to what is currently called *environment awareness*.

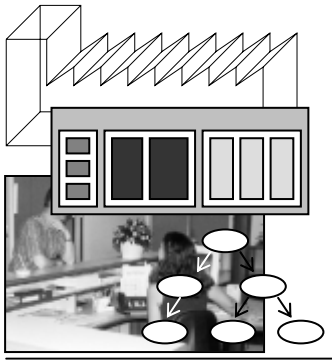
[Papazoglou and Heuvel, 1999] explained that it is necessary to have an understanding of the organisational environment, its goals and policies, so that the resulting systems will work effectively together with human agents to achieve a common objective. Many formal languages exist to describe a model, see [Gastinger and Szegheo, 2000] for an overview of enterprise modelling approaches. In CoMMA, we decided to decouple the modelling language from the formalism by doing ontology-based modelling. The methodology IDEF5 in the beginning of the 90s proposed also to develop ontologies for enterprise modelling. The modelling process in that case is split in two:

1. the design of an ontology that will provide natural customised yet unambiguous vocabulary to express the models.
2. the implementation of the model using a formalism supporting our ontology.

To benefit from the XML standard technology assets, we decided to use the RDF Schema and RDF language to describe our ontology and implement our models: organisational entities and people are annotated. This choice enables us to base our system on the W3C recommendations that benefit from all the web-based technologies for networking, display and navigation, and this is an asset for the integration to a corporate intranet environment.

As I said, a corporate memory has a delimited and defined context, infrastructure and scope: the corporation. In the corporate context, we can more precisely identify the stakeholders (e.g.: information providers, information seekers); moreover, the corporate community shares some common global views of the world (e.g.: company policy, best practices) and thus an ontological commitment is conceivable to a certain extent.

Based on this ontology, we describe the organisational state of affairs including the enterprise structural model and the users' profile. The expression "state of affairs" was imported on purpose from the field of ontology engineering. It refers to the general state of things, the combination of circumstances at a given time, that will be subject to a description in the conceptual terms provided by the ontology. The corporate description takes the shape of annotations about the organisational entities (divisions, groups, etc.) and their relations (manage, include, etc.). Figure 38 echoes the example of cubes given in Figure 4 (page 67) adapting it to our application here.



Organizational Entity (X): The entity X is or is a sub-part of an organization.
Person (X): The entity X is living being pertaining to the human race.
Include (Organizational Entity: X, Organizational Entity / Person Y): the organizational entity X includes Y as one of its members.
Manage (Person: X, Organizational Entity: Y): The person X watches and directs the organizational entity Y

Person(Rose)
Person(Fabien)
Person(Olivier)
Person(Alain)
Organizational Entity(INRIA)
Organizational Entity(Acacia)
Include(INRIA, Acacia)
Manage(Rose, Acacia)
Include(Acacia, Rose)
Include(Acacia, Fabien)
Include(Acacia, Olivier)
Include(Acacia, Alain)

a - Reality **b** - Ontology **c** - State of Affairs

Figure 38 Reality, Ontology and State of Affairs in the context of an organisational memory

Likewise, the users' profile captures all aspects of the user identified as relevant for the system behaviour. It contains administrative information and the user's preferences that were directly made explicit: such preferences can go from interface customisation to topic interests. The user's profile also positions the user in the organisation: role, location and potential acquaintance network. In addition to explicitly stated information, the CoMMA system derives information from the usage made by the user. It collects the history of visited documents and possible feedback from the user, as well as the user's recurrent queries, failed queries, and from this it can learn some of the user's habits and preferences. These derived criteria can then be used for interface purposes or push technology. Finally, the profiles enable to compare users, to cluster them based on similarities in their profiles and then use the similar profiles to make suggestions (techniques of collaborative filtering).

To summarise, the three stages in building a corporate semantic web are:
- We applied knowledge engineering techniques for data collection in order to provide the conceptual vocabulary detected as needed in the scenarios. We specified the corporate memory concepts and their relationships in an **O**ntology (O'CoMMA) and we formalised the ontology in RDFS.
- We used the conceptual vocabulary of the ontology and the results from data collection to develop enterprise and user models in order to describe the organisational **S**tate of affairs. These models are implemented in RDF and instantiate the RDFS ontology description.
- We structure the corporate memory by writing RDF **A**nnotations on the **D**ocuments: these annotations instantiate the RDFS ontology description and refer to the state of affair.

The memory structure is illustrated in the sequence of small illustrations shown in Table 9.

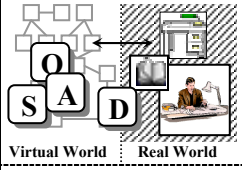
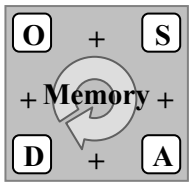
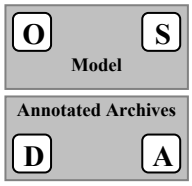
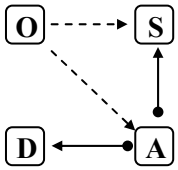
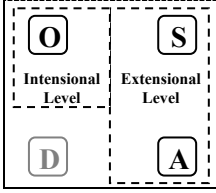
	The **O**ntology, the **A**nnotations and the **S**tate of Affairs form a virtual world capturing the aspects of the real world that are relevant for the system.
	The memory is composed of the **D**ocuments, their **A**nnotations, the **S**tate of Affairs (user profiles and organisation model) and the **O**ntology. The whole follows a prototypical lifecycle, evolving and interacting with each other.
	The **O**ntology and the **S**tate of Affairs form the model on which is based the structuring of the memory. The archive structure relies on the **A**nnotations of the **D**ocumentary resources.
	The **A**nnotations and the **S**tate of Affairs are formalised using the conceptual vocabulary provided by the **O**ntology. The **A**nnotations reference the **D**ocuments (e.g. report http://www...) and the objects of the **S**tate of Affair (e.g. written by Mr. X for the division ABCD)
	The **O**ntology defines modelling and annotation primitives at the intensional level. The **S**tate of Affairs and the **A**nnotations instantiate these primitives describing models and annotation of the memory at the extensional level.

Table 9 Commenting the Memory structure

6.3 CORESE: COnceptual REsource Search Engine

Keyword-based search engines work at the term level and one of the reason for introducing ontologies in CoMMA is that they are a means to enable software to reason at the semantic level. To manipulate the ontology, the annotations, and infer from them, my colleagues developed CORESE [Corby *et al.*, 2000] [Corby and Faron-Zucker, 2002], a prototype of search engine enabling inferences on RDF annotations and information retrieval from them. RDF(S) was influenced by the graph data models and we manipulate it as a restriction of Sowa's Conceptual Graphs (CGs) [Sowa, 1984].

CORESE combines the advantages of using (a) the RDF(S) framework for expressing and exchanging metadata, and (b) the query and inference mechanisms available for Conceptual Graph formalism [Sowa, 1984].

A Conceptual Graph is a bipartite graph, where every arc links a concept node and a conceptual relation node. Both concept and conceptual relation have a type that can be either primitive or defined by a monadic lambda expression. Concept and relation types are organised in two hierarchies which are sets of type labels partially ordered by the subtype relation. Figure 39 shows that RDF schemas and statements can be translated, respectively, into type hierarchies and directed bipartite graphs. The RDF triple model only supports binary relations, thus RDF annotations generate Conceptual Graphs with dyadic relations of primitive type and the signature of relations are derived from the *domain* and *range* constraints.

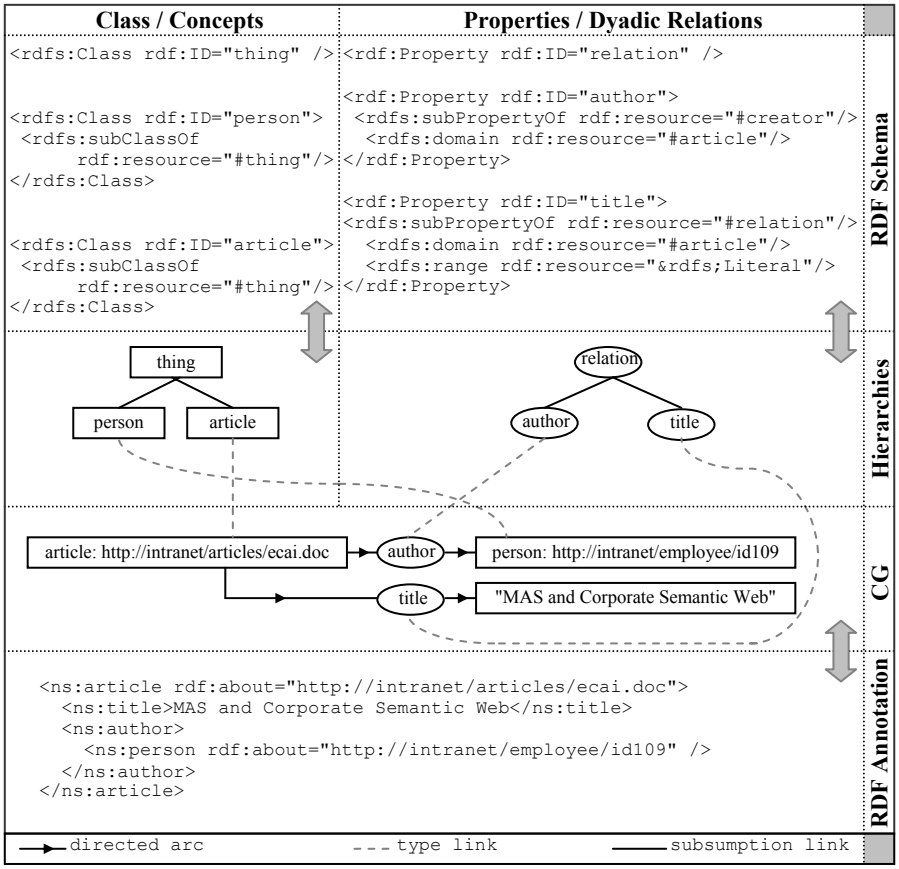


Figure 39 Mapping RDF(S) and CG

Type hierarchies are at the heart of ontologies for information searching and communication since they support inferences used when querying: specialisation, generalisation and identification. The Conceptual Graph projection operator is well adapted to annotation retrieval as it performs matching taking into account specialisation of relations and concepts. Both precision and recall are thus improved:

- a query looking for *documents* concerning *vehicle* securities will also retrieve *documents* concerning *car* securities, *reports* on *train* securities, etc. since all these concepts are specialisations of the concepts used for querying; the answers retrieved are relevant thus the recall is improved compared to a purely term-based retrieval.
- a query looking for *documents* concerning *pipe* diameters, with pipe being an instance of the concept of "tubes", will not retrieve documents concerning the diameter of pipe as smoking tools; the precision of the query is thus improved.

The query language used in CORESE is RDF with variables (prefixed by a question mark) to indicate the pattern to be found, the values to be returned and the co-references constraints. Regular expressions and comparators are used to constrain literal values and additional operators are used to express disjunction and negation.

Comparators are [Corby and Faron-Zucker, 2002]:

- numeric ordering and lexicographical order: *a* <= *b* (*a* lesser than or equal to *b*) , *a*<b (*a* lesser than *b*), *a*>=*b* (*a* greater than or equal to *b*), *a*>*b* (*a* greater than *b*)
- string comparison: *a* = *b* (*a* and *b* are identical), *a*~*b*(*a* contains *b*, ignoring case), *a*^*b* (a starts with b)
- type comparison: *a*<:*b* (*a* strict subtype of *b*), *a*<=:*b* (*a* subtype or equal to *b*), *a*=:*b* (*a* and *b* are identical types), *a*>=:*b* (*a* super type or equal to *b*)

In Figure 40 are presented three examples of queries. The first one asks for all the persons interested in computer science, with their name. The second one looks for all the documents concerning Java progamming that are not reports, and requires their title and the ISBN if it is available. Finally, the third query is a query on the ontology and asks for all the classes which label contains reports, and their super-classes.

```
1   <CoMMA:Person CoMMA:Name='?Name'>
2    <CoMMA:IsInterestedBy><CoMMA:ComputerScienceTopic/></CoMMA:IsInterestedBy>
3   </CoMMA:Person>
4
5   <CoMMA:Document rdf:about='!<=:Report'>
6     <CoMMA:Title>?title</CoMMA:Title>
7     <CoMMA:HasForISBN>?isbn $option</CoMMA:HasForISBN>
8     <CoMMA:Concern><CoMMA:JavaProgramingTopic/></CoMMA:Concern>
9   </CoMMA:Document>
10
11  <rdfs:Class rdfs:label='~report' rdfs:subClassOf='?Parent'/>
12
```

Figure 40 Examples of queries in CORESE

The general process of CORESE is shown in Figure 41. CORESE maps RDF statements to a base of Conceptual Graph facts, the class hierarchy defined in an RDF schema is mapped to a concept type hierarchy in the Conceptual Graphs formalism and the hierarchy of properties described in the RDF schema is mapped to a relation type hierarchy in Conceptual Graphs. The concept type hierarchy and the relation type hierarchy constitute what is called a support in the Conceptual Graph formalism: they define the conceptual vocabulary to be used in the Conceptual Graphs for the considered application. When submitted an RDF query is translated into a CG that is projected onto the CG base to isolate any matching graph and extract the requested values that are then translated back into RDF.

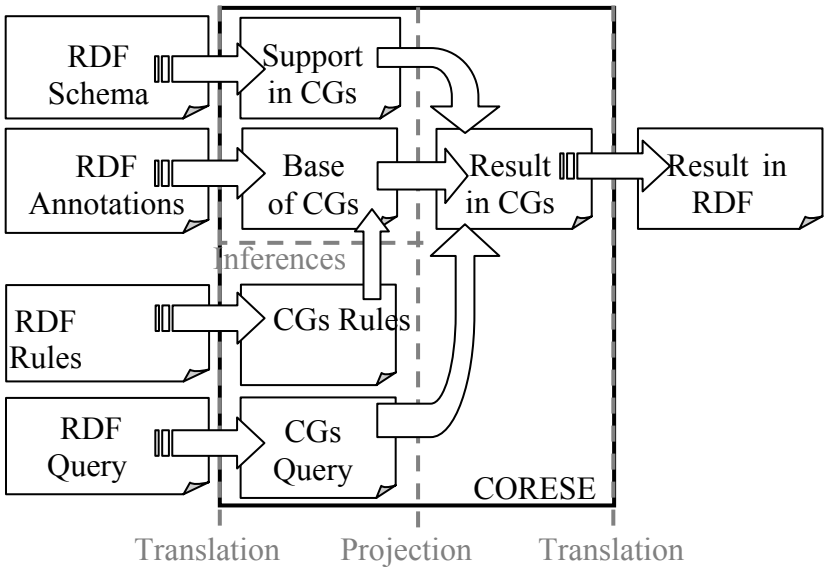


Figure 41 Process principle of CORESE

We saw that axiomatisation in an ontology can lead to rules that enable to deduce new knowledge from existing one. RDF Schema does not provide such a mechanism does not provide the expressiveness necessary to capture axioms or rules. Hence, as shown in Figure 41, CORESE proposes an RDF Rule extension to RDF & RDFS and an inference engine based on forward chaining production rules. [Corby and Faron-Zucker, 2002]

These rules are inspired by conceptual graphs rules [Salvat and Mugnier, 1996] where a rule $G_1 \Rightarrow G_2$ applies to a graph G if there is a projection from G_1 to G *i.e., G* contains G_1 or a specialisation of G_1. If it is the case then G is augmented with G_2. Replacing graphs by annotations, CORESE represents $A_1 \Rightarrow A_2$ with the following language:

```
1   <cos:rulebase xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:s="http://www.inria.fr/acacia/schema#"
       xmlns:cos="http://www.inria.fr/acacia/corese#" cos:name='SampleBase'>
2
3     <cos:rule cos:name='SampleRule'>
4       <cos:if>
5                          ⎫
6         ...              ⎬ A1
7                          ⎭
8       </cos:if>
9       <cos:then>
10                         ⎫
11        ...              ⎬ A2
12                         ⎭
13      </cos:then>
14    </cos:rule>
15
16  </cos:rulebase>
```

Figure 42 Rule language of CORESE

The annotation pattern A_1 is described like a query and A_2 is described using the variables used in A_1 to define the additive and/or specialising pattern.

Moreover, algebraic properties of relations may be denoted using an extension of RDFS; these properties are: transitive, symmetric and reflexive. In Figure 43 is an example of the 'Related Document' property declared transitive and symmetric.

```
1   <rdf:Property rdf:ID='RelatedDocument'>
2     <rdfs:domain rdf:resource='#Document'/>
3     <rdfs:range rdf:resource='#Document'/>
4     <cos:transitive>true</cos:transitive>
5     <cos:symmetric>true</cos:symmetric>
6   </rdf:Property>
```

Figure 43 Example of transitive and symmetric property

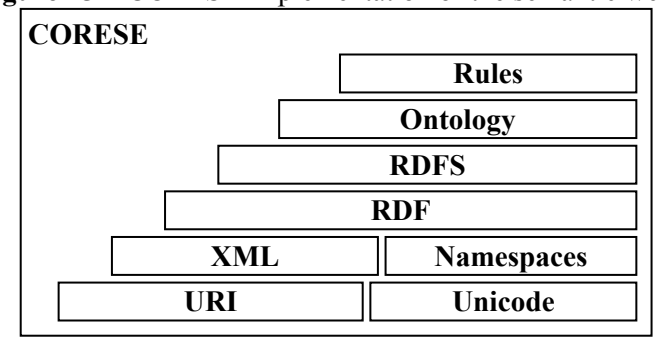
Finally, inverse relations may be expressed as shown in Figure 44.

```
1   <rdf:Property rdf:ID='include'>
2     <rdfs:domain rdf:resource='#Organization'/>
3     <rdfs:range rdf:resource='#Person'/>
4     <cos:inverse rdf:resource='#MemberOf'/>
5   </rdf:Property>
```

Figure 44 Example of inverse property

Thus, building on top of available packages (XML parser, RDF parser, Conceptual graphs platform, etc.) CORESE provides a semantic search engine and an API rising up to the sixth layer of the semantic web stack (Figure 45). I used it to implement the behaviour of the agents manipulating the ontology and the annotations.

Figure 45 CORESE implementation of the semantic web stack



CORESE is a lightweight software providing a standalone semantic search engine in the shape of a server or an API to build other software needing RDF(S) handling capabilities.

6.4 On the pivotal role of an ontology

The ontology is the keystone of the CoMMA system since it provides the building blocks for models, annotations and agent messages, with their associated semantics.

The ontology-based approach was motivated by needs and observations that arose in both scenarios and I identified the three roles it plays in the CoMMA system:

- First, the ontology is a key component of the corporate memory that captures a knowledge about the company and the application domain. This knowledge may be useful, for instance, to a new employee trying to familiarise himself with the organisation terms and policies.
- Second, it is central to multi-agent systems where agents need to rely on a shared conceptual vocabulary to express and understand the messages they exchange. In both scenarios, the agents must be able to understand each other in order to co-operate and assist the different users properly.
- Finally, in both scenarios, the heterogeneous documents are manipulated thanks to their semantic annotations and retrieved through queries. Annotations and queries are based on the same shared ontology that provides a consensual conceptual vocabulary to expressed them so that they can be compared and matched.

In the application scenarios it is vital to ensure the quality of communication between and among artificial and human agents. An ontology can play the role of the needed semantics grounding for communication and representation of knowledge. An ontology is a new object of Artificial Intelligence that recently came to maturity and a powerful conceptual tool of Knowledge Modelling. It provides a coherent base to build on, and a shared reference to align with, in the form of a consensual conceptual vocabulary, on which one can build descriptions and communication acts. By making explicit the so often implicit conceptualisation of the world, ontologies enable us to capture consensus, formalise knowledge and exchange it. Even if it is only partial, this explicit conceptualisation enables us to simulate intelligence in artificial actors of information spaces.

In CoMMA I developed only one ontology; agents and users exploit different aspects of the same ontology. In the next section, I describe the process of engineering this ontology called O'CoMMA (ontology of CoMMA).

7 Engineering the O'CoMMA ontology

In this chapter I explain step by step how the ontology O'CoMMA was built. The following sections concentrate on the approach and the design rationale while the following chapter presents and discusses the results of this experience.

The design of O'CoMMA has been influenced by (1) the return on experience and the analysis on TOVE and the Enterprise Ontology done by Uschold and Gruninger [1996], (2) the comparison and elements of methodology presented in [Fernandez *et al.*, 1997] and [Gómez-Pérez *et al.*, 1996] and (3) the work done on theoretical foundations of ontologies by Bachimont [2000] and Kassel *et al.* [2000].

In the following parts, I present the approach, my return on experience, and my expectations for the evolution of the ontology engineering field. First I shall explain the data collection process and why a scenario-based analysis is interesting in driving this collection. Then, I shall discuss the reuse of existing ontologies and the use of expertise sources external to the organisation. Finally, I shall describe the refinement process starting from terminological analysis and informal intermediary representations and gradually moving to formal representation and axiomatisation.

7.1 Scenario analysis and Data collection

Several techniques exist for data collection that feeds the whole modelling process. We essentially used three of these techniques: semi-structured interview, observation and document analysis coupling them with an analysis phase based on scenarios (Stage 1 in Figure 46 page 215).

7.1.1 Scenario-based analysis

Storytelling reveals meaning without
committing the error of defining it.
— Hannah Arendt

Scenarios are textual descriptions of the organisational activities and interactions concerning the intended application. Following [Caroll, 1997] we used scenarios to capture in a natural and efficient way the end-users' needs in their context. This is vital for a symbiotic integration of the system in the work environment. As I indicated previously, in CoMMA, we chose to focus on two scenarios:
- *New employee integration* assistance: how can we accelerate and facilitate the integration of a new employee in the company?
- *Technology monitoring* support: how can we support the task of identifying, annotating and broadcasting relevant technological news in a company?

The main advantages we recognised when using scenarios for CoMMA were:
- Scenarios enabled us to *focus on the specific aspects of knowledge management* involved in our case.
- They helped us *capture the whole picture* and they enabled us to view the system as a component of a possible knowledge management solution for a company.
- They represented a *concrete set of interaction sequences* with the corporate memory, understandable and accessible to all the stakeholders. Thus they were a perfect start to build formal use cases and interaction diagrams.
- They provided a *framework to check* up on every new idea, every contribution.

Scenario analysis led us to define a table (c.f. Table 10) suggesting key aspects to be considered when describing a scenario. It provided suggestions as for what are the aspects to be investigated, what to look for and what could be refined when describing a scenario. It helped us define the scope of our intervention and thus the scope of the ontology: the conceptual vocabulary had to provide the expressiveness required by the interactions, between the system and its environment, described in the scenarios [Breuker and Van de Velde, 1994].

Ontologies like TOVE [TOVE, 2000], or Enterprise Ontology [Uschold *et al.*, 1998] have developed, for instance, a view concerning the processes and workflow whereas in CoMMA we did not model those aspects since we do not exploit them in our scenarios. However our part on the document aspect, which is central to a documentary corporate memory, is much bigger and detailed than in TOVE or Enterprise ontologies. I defined the criteria used for evaluating the ontology coverage as being its exhaustivity, specificity, and granularity; the scenarios provide a testbed of this coverage to check if for the envisaged usage the exhaustivity, specificity, and granularity of the ontology are sufficient.

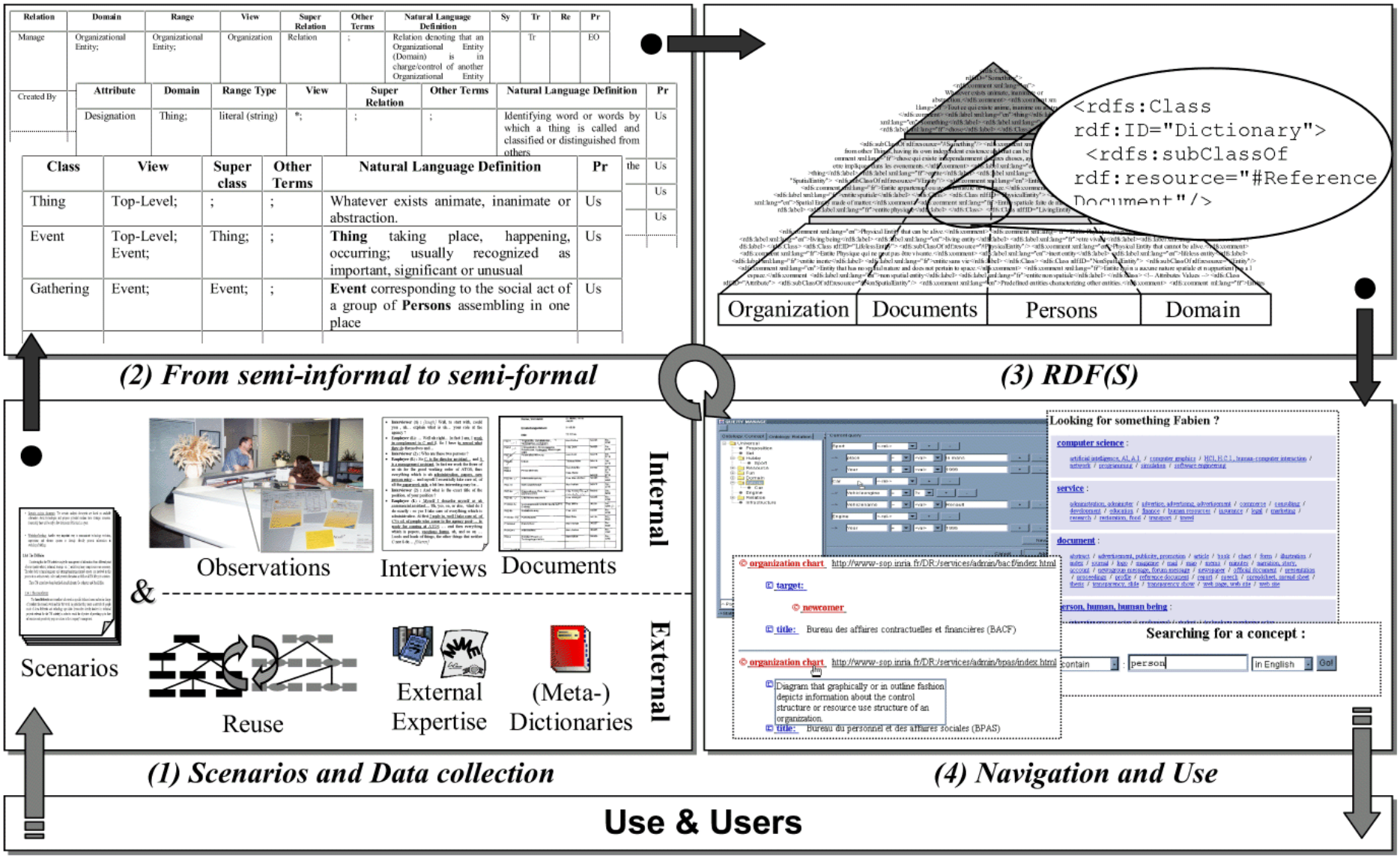


Figure 46 Ontology design process

<table>
<tr><th>Characteristics</th><th>Representation</th><th colspan="2">Facets</th></tr>
<tr><td rowspan="6">Goal

Scenario Before / Scenario After

Scope

Scenario / Sub-Scenario

Generic / Specific Example, Illustration

Relevance life-time

Exceptions
Counter examples</td><td rowspan="6">Textual
Graphical

Informal
Formal (UML)

<u>NB:</u> In one scenario description, several types of representations may be used.</td><td>Actors</td><td>Profile
Role
Individual goal
Task
Action
Interaction</td></tr>
<tr><td>Resources</td><td>Nature
Services
Constraints</td></tr>
<tr><td>Logical & Chronological</td><td>Processes
Decomposition
Sequential / Parallel /Non deterministic
Loops & Stop conditions
Alternatives & Switches
Compulsory / Optional</td></tr>
<tr><td>Flows</td><td>Inputs
Outputs
Paths</td></tr>
<tr><td>Functionalities & Rationale</td><td>Functionalities description
Motivation, necessity
Advantages & Disadvantages</td></tr>
<tr><td>Environment</td><td>Internal
Organisation
Acquaintance
External</td></tr>
</table>

Table 10 Scenario template table

The table gives different aspects to be mentioned or looked for during data collection, but it is neither a list of compulsory fields to be populated, nor a restrictive list, and it must not become a bias. It is a starting point to initialise and support data collection, avoiding an excessively free, unstructured and unfocused collection leading to information overflow and irrelevance. This table reduces the risks of missing some important aspects of the scenario.

Scenario analysis produced traces: in our case, scenario reports. As illustrated in the sample given in Figure 40, these reports are extremely rich story-telling documents the content of which is highly relevant for the usage envisaged in the scenarios; therefore, these documents are themselves good candidates to be included in the corpus of the terminological study that will be one of the following steps in designing the ontology. Scenarios were our first step in data collection and their grid is used during the collection and analysis to guide and focus both activities.

"*Scenario analysis documents*: The scenario analysis documents are based on available information about technologies and propose potential medium term strategic scenarios. Reasonably there will be only a few documents of this kind in a year.

Workshops/briefings: Another very important way to communicate technology evolution, impressions and discuss opinion is through directly present information in workshops/briefings."

(...)

"**3.3.5 The TM Roles**

Considering that the TM activities imply the management of information from different points of view (market-related, technical, strategic. etc.) multidisciplinary competencies are necessary. Therefore, both technical engineers and strategic/marketing-oriented experts are involved in this process to co-ordinate work, collect and present information and follow all TM lifecycle activities.

Three TM actors have been identified: Area Referents, Co-ordinators and BackOffice.

3.3.5.1 The area referents:

The Area Referents are researchers who work in specific technical areas and are in charge of correlate the research work and the TM work. In particular they create a network of people made of Area Referents and technology specialists (researchers directly involved in technical projects relevant for the TM activity) in order to reach the objective of providing up-to-date information and proactively propose actions to the company's management."

Figure 47 Extract from scenario report written by an end-user following the previous grid (double underline shows the interesting notions that can be found in this extract)

In Figure 47, we underlined the candidate terms interesting for modelling. As a toy example, we could in particular deduce from this passage that three technology monitoring roles were identified and a partial sub-tree could be generated as in the toy example of Figure 48.

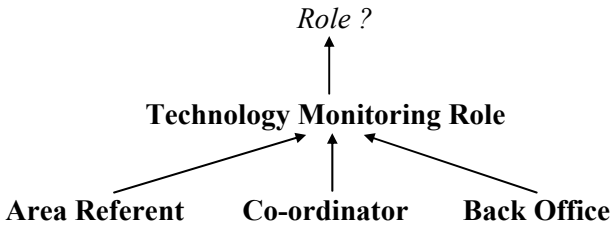


Figure 48 Example of a taxonomy of concepts

Of course, consistency checks are to be carried out with regard to the modelling point of view adopted for the scenarios (relevant aspects, conceptualisation used by end-users, coverage needs) and the existing work on the ontology. This extension would thus be refined and then grafted to the rest of the ontology. This addition is typically in a bottom-up perspective, and completeness and coherence require to trigger new tasks in the other perspectives (top-down and bottom-up) e.g.: is this list of TM Role complete? Does the role core concept exist? Are there ontologies dealing with that subject?...).

Scenarios grids and reports are used to initialise, focus and evaluate the whole knowledge modelling and the whole application design.

7.1.2 Semi-structured interviews

Semi-structured interviews were carried out in three steps:
- A *free opening* discussion where the dialog was initiated with broad questions concerning the tasks and roles of the people interviewed. If needed this spontaneous expression could be kept running using short questions.
- *Flashbacks* and clarifications on specific questions prepared before the interview.
- A *self-synthesis* in order to make interviewees synthesise, summarise and analyse themselves what they said and make them conclude. The generalisation and grouping they performed in this last part were especially interesting for the ontology structuring.

We carried out an interview with a newcomer at ATOS (an industrial partner of the consortium). Figure 49 is extracted from this interview retranscribed and analysed by my colleague Alain Giboin. As an example, we can find in it interesting information about documents manipulated (e.g.: "vacation forms"), their use and the tasks they are linked to (e.g. "to issue an order"). During the interview, we also discussed the importance of the acquaintance network in her day-to-day activity and we derived from this the necessity of the structural organisation model of her company to support the scenario; this has consequences both on the coverage of the ontology and on the models to be instantiated.

Interviewer (1): *[laugh]* Well, to start with, could you, uh… explain what is uh… your role at the agency ?

Employee (L): … Well uh right… In fact, I am, I work in complement to C and S. So I have to reread what they do themselves and…

Interviewer (2): Who are these two persons ?

Employee (L): So C, is the director assistant… and S is a management assistant. In fact, we work the three of us uh for the good working order of ATOS, thus everything which is uh administration, papers, new person entry… and myself I essentially take care of, of all the paperwork side, a bit less interesting maybe…

Interviewer (2): And what is the exact title of the position, of your position ?

Employee (L): Myself I describe myself as uh commercial assistant… Uh, yes, so, or else, what do I do exactly: so yes I take care of everything which is administrative. At first I reply to, well I take care of, of C*vs.* of, of people who come to the agency post-… to apply for coming at ATOS … and then everything which is papers, vacations forms, uh, and so on … Loads and loads of things, the other things that neither C nor S do… *[Silence]*

Figure 49 Extract from an interview

During an interview, we can also detect specific aspects that will have repercussions on the whole system specifications. For instance in the previous example, the role described by the interviewee "commercial assistant" is different from her official role "secretary". Since the definition this newcomer had of her role and position was different from what was stated in the official organisation chart, it is important to have and exploit user profiles in the CoMMA solution in addition to the enterprise model, to allow the system to adapt to the specificity of each user. The characteristics of these profiles have to be captured and this places a new requirement on the ontology.

The two major problems with this kind of collection and analysis is that it is time consuming and it is prone to overload designers with details.

7.1.3 Workspace observations

Another data-collection technique we applied was the observation. The observation can be about people, the way they work (with or without letting them comment their activity), on a real task or on a simulated scenario, in real time, recorded or based upon traces of the activity. It could also be focused on chosen indicators (e.g. documents manipulated, desk organisation, etc.). Depending on the actor and the scenario, the interesting situation may be very different (e.g. a newcomer looking for information, a mentor explaining a technology, an observer commenting, etc.).

We observed the desk of the newcomer we interviewed, as a lot of documents in the company go through her since she is an assistant. Figure 50 shows four pictures of her working place.

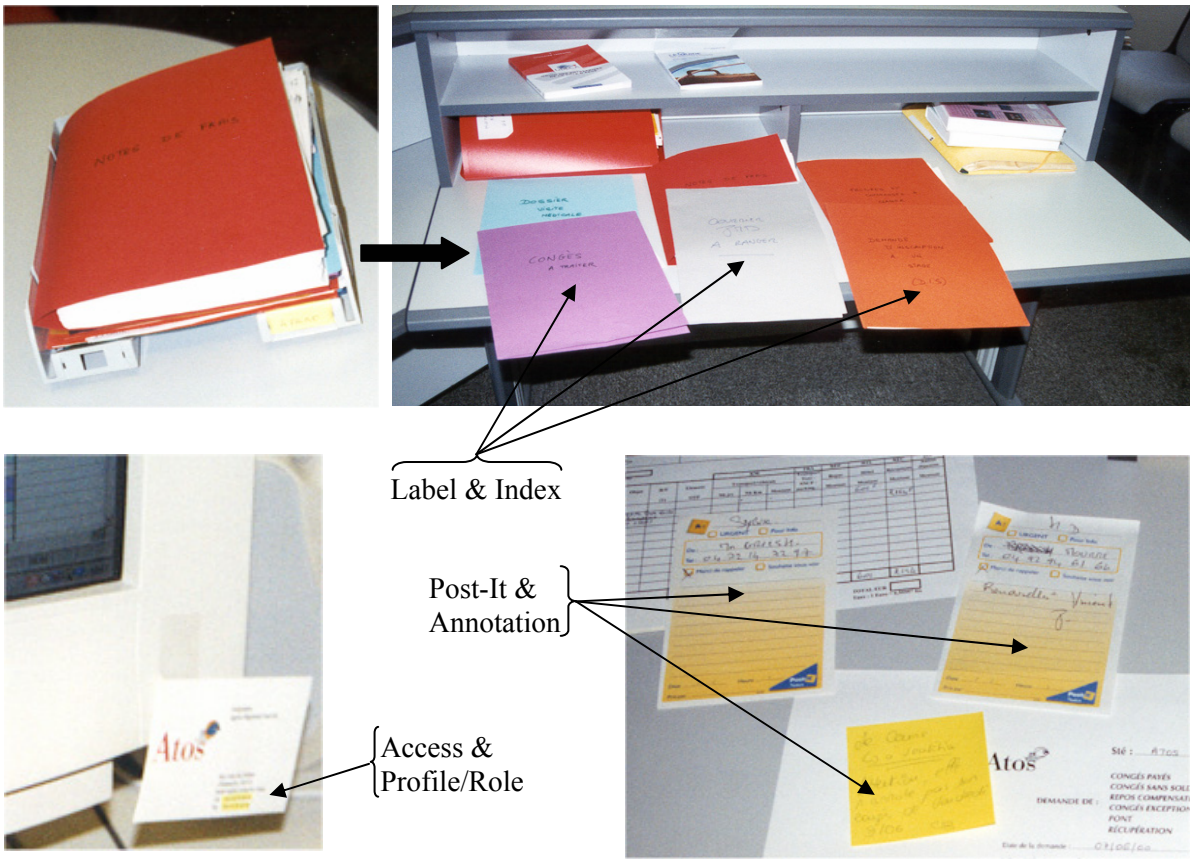


Figure 50 Observing working documents

Observations reveal that she was using two criteria to label the files: the type of documents (e.g. "vacation forms") and the process to be done on these documents (e.g. "to be signed"). Another observation shows different types and uses of annotations on documents using post-it (e.g. targeted people, deadline, how to fill and use a form); these are clues on the type of indexing annotations end-user may want to make. Finally, we also noticed that there are documents (in our case the company name card with the phone number and the fax number) that she does not want to sort, and put away with others (phone book, address book). Because of her activity, she wants to access it at the first glance when needed; in a system this would mean the ability to index/bookmark some documents so as to access them 'at the first click'.

These observations helped us understand what type of vocabulary was needed for the annotations, but not all the observations are relevant for scenarios or lead to feasible specifications. Once again, great care has to be taken to focus on the scenario specific and relevant aspects during data-collection.

7.1.4 Document Analysis

The last type of data-collection techniques we used is the document collection and analysis. The gathering of typical documents is vital for a project that focuses on information retrieval in a corporate documentary memory. The purpose is to systematically collect documents involved in and relevant for the considered scenarios: lexicon, terminology, reports, graphical documents, forms, organisation charts, etc. A good practice is to collect both empty and filled forms so as to see their use, and also to 'follow a document' to its track in the organisation.

An example of such document is the new employee route card of T-Nova (Figure 51): it describes what to do, where to go, who is the contact, how to contact the person, and what is the order of the tasks to be carried out by a new employee once arrived.

Name, Vorname: Dr. Müller, Heinz Jürgen

Einstellungsdatum: 01.05.98

DSt: TZ FE14k

PSZ-4	Angestellte, Gehaltskonten, Vermögensw. Leistungen	Herr Kottke	54/425	Tel.: 3780
PSZ-8	Umzugskosten, Trennungsgeld, Reisekosten, Kindergeld, Kindertagesstätte	Frau Deml	54/529	Tel.: 2724
PSZ-3a	Personalbuchführung	Herr Grall	54/418	Tel.: 2718
PSZ-3b	Urlaub	Thomas Mohr	54/418	Tel.: 2719
PSZ-1 ~~PSZ-2~~	Personaleinsatz	Herr Schröter ~~Frau Hierer~~	54/420 ~~54/418~~	Tel.: 2711 Tel.: ~~2715~~
PSZ-5a	Arbeitszeitregelung	Frau Maul-Gottaut	54/340	Tel.: 5954
PSZ-10	Wohnungsfürsorge	Herr Halfen	54/511	Tel.: 2733
PSZ-9a	Krankenkasse, Post-, Spar- und Darlehnsverein	Frau Polzer	54/510	Tel.: 8252
PSZ-6b	Unternehmensausweis	Frau Weingart	54/438	Tel.: 1238
P183DA-1a	Vorübergehende Unterbringung BZ/FH Dieburg	Frau Müller	36/251	Tel.: 8291
PSZ-9b	Sozialbetreuung	Frau Loos	54/531	Tel.: 2728
P183-DA-11b	Parkerlaubnis	Herr Scior	34/123	Tel.: 6623
P184DA-2	Brandschutz	Herr Harsch	36/227	Tel.: 6620
P184DA-1	Arbeitsschutz	Herr Anders	36/226	Tel.: 7697
PSZ-4	Sonstige Fragen zu Tarifangelegenheiten	Herr Kottke	54/425	Tel.: 3780

01.05.98

Figure 51 Deutsch Telekom new employee route card

This document gives an idea of the stakeholders involved in the new employee scenario, and it reveals the existence of a process of integration and the vocabulary needed to describe the whole state of affairs.

This document is also a good example of a lethal problem in knowledge acquisition: the problem of access to information. Here the natural language used for documents (German) may not be comprehensible to the knowledge engineer. This problem has been raised in several international

projects and is time-consuming and money-consuming (train native speakers to analyse it themselves, translate documents,...). Another form of obstacle to information access we encountered was the security restriction on documents. *An answer to access problem is to train some end-users to do the analysis themselves*. It can be time-consuming, but it is also a good point to ensure the system maintenance at running time.

In order to scale-up the process to a large amount of documents, Natural Language Processing tools can be used to scan corpora such as the intranet or large samples of documents. A detailed experience was carried out in our team at Renault car constructor [Golebiowska, 2002].

These tools for text analysis have not been used for O'CoMMA, but are extremely helpful since textual documents are the most common type of documents available. However some documents use graphical conventions that are meaningful, but not automatically exploitable (e.g. the graphical layout of the organisation chart in Figure 52). For these documents, a 'manual' analysis is still necessary whereas for textual documents a semi-automatic analysis can be envisaged.

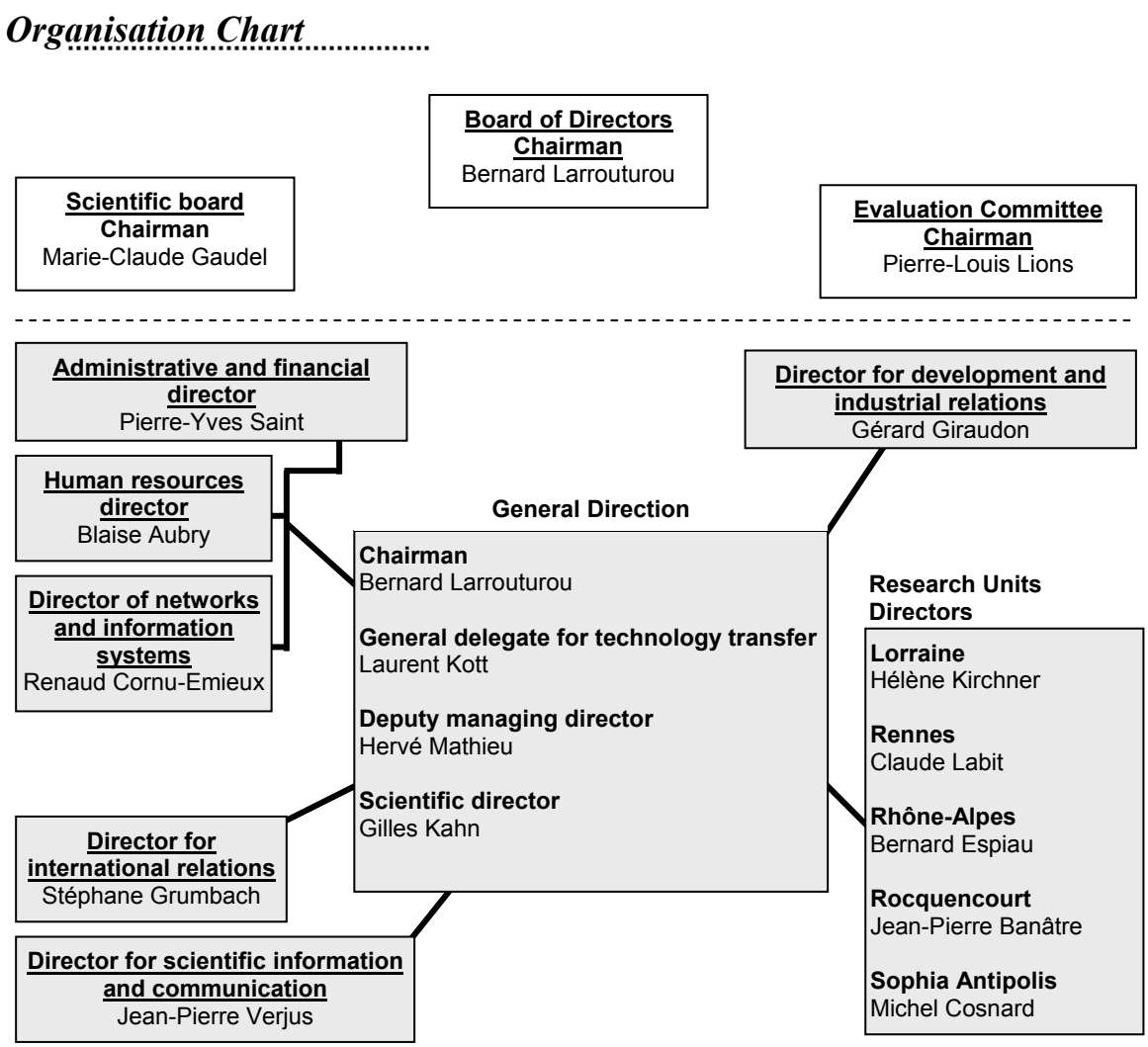


Figure 52 Organisation chart of INRIA

7.1.5 Discussion on data collection

In CoMMA we did not use questionnaires for data collection; however we used them during the trial and evaluation to organise the tests, to structure the use of the first prototypes, and to get feedback from the users.

A recurrent point in the method explained here is the time-consuming and effort-consuming aspects of these methods. Some of them may be partially automated like text-analysis, but most of them are manual. In fact, the key point I am making here is not to extensively use data-collection techniques, but use them in a focused fashion, target subject and objects profiles (e.g. for new employee scenario we chose a new assistant thus combining newcomer and document-worker aspects) and above all combine them to cross-check and crossbreed results.

Collection and analysis are time-consuming and resource-consuming yet they enable the designers to discover implicit and hidden aspects of the conceptualisation. Scenario-based analysis can be used to guide and focus the whole collection, analysis and design process. To avoid access restriction problems, to improve the capture of user's conceptualisation and prepare maintenance, it is interesting to train end-users to carry out these tasks themselves as much as possible.

7.2 Reusing ontologies and other sources of expertise

Data collection methods were extremely time-consuming. In order to speed-up the process we decided to also study and reuse existing ontologies whenever possible:
- Enterprise Ontology [Uschold *et al.*, 1998]
- TOVE Ontology [TOVE, 2000]
- Upper Cyc® Ontology [Cyc, 2000]
- PME Ontology [Kassel *et al.*, 2000]
- CGKAT & WebKB Ontology [Martin and Eklund, 2000]

The reuse of ontologies is both seductive (it should save time, efforts and would favour standardisation) and difficult (commitments and conceptualisations have to be aligned between the reused ontology and the desired ontology). For this last reason, the existing ontologies cited above have not been imported directly or translated automatically. We found that the best way to reuse them was to analyse their informal version as a textual document collected. This was possible only because these ontologies are very well documented. Divergences in the objectives, the contexts of modelling and uses, between these ontologies and O'CoMMA (the end-users and application scenarios were indeed different) led us to think that *no automatic import was possible but rather a human-supervised import and a provision of inspiration*. However natural language processing tools, such as the ones used in the approach proposed by [Aussenac-Gilles *et al.*, 2000], could have helped the analysis. Moreover, translator between formal languages could have eased reuse.

In addition to these contributions, we had to consider other informal sources. Some specific sources of expertise helped us structure upper parts of some branches. For instance, we used thoughts from the book "Using Language" from Herbert H. Clark for structuring the document branch on representation systems, providing an expertise on semiotic that was needed and could not be gained through previous data collection techniques (the interviewed stakeholders were not expert in semiotics, and the collected documents did not explicitly contain that information either). Other very specific standards enabled us to save time on enumerating some leaves of the ontology. For instance, the MIME standard was an excellent input for electronic format description and the Dublin Core suggested most common global properties of documents.

The systematic use of dictionaries or available lexicons is good practice. In particular, the meta-dictionaries have proved to be extremely useful. They give access to many dictionaries (some of them specialised in specific fields e.g. economy, technology...) and therefore, they enabled us to compare definitions and identify or build the definition corresponding to the notion we wanted to introduce. We made extensive use of the meta-dictionary OneLook[1] that enabled us to produce the first English expressions of intensions of the O'CoMMA concepts.

Scenarios were used to prune the contributions to the ontology. These scenarios captured the scope needed for the ontology and a shared vision of the stakeholders: thus they helped to decide whether a notion candidate for reuse was relevant or not. For instance, in O'CoMMA, we did not reuse the "ownership" relation of the Enterprise Ontology since this relation was useless in our scenarios.

Reuse should be pursued wherever possible, even though in the best case, only semi-automatic techniques can be realistically envisaged. Moreover, *no organisation is an island* and relevant resources external to the organisation have to be included in the collection and analysis process; *knowledge is holistic and its collection is bounded by the scope of the scenarios not by the organisational boundaries*.

[1] http://www.onelook.com

7.3 Initial terminological stage

"Think simple" as my old master used to say
meaning reduce the whole of its parts into the
simplest terms, getting back to first principles.
— Frank Lloyd Wright

The candidate terms denoting concepts that appeared relevant in the application scenarios were candidates for the terminological analysis. Synonymous terms were selected too, and related terms were considered in a chain reaction. For instance, if we consider the terms *document*, *report*, and *technological trend analysis report* involved in the technology monitoring, their candidacies to terminological analysis were linked. The starting point could be the term *document* due to a study of the existing top ontologies, or the term *report* identified during the interview of the ATOS newcomer or finally, the term *technological trend analysis report* encountered when collecting documents for the technology monitoring scenario (e.g.: "*technological trend analysis* report entitled 'Capitalising WAP experiences for UMTS transition' ").

Candidate terms were organised in a set of informal tables – that form a semi-informal data collection structure. INRIA proposed definitions in natural language for each term, building a lexicon. This first terminology was presented to members of the CoMMA consortium and low-level extensions (terms and definitions) were proposed by the industrial partners, for instance:

AREA REFERENT: Observer responsible for an expertise area and who has a group of contributors observers to manage.

It was clearly interesting to have a continuous collaboration between, on the one hand, a knowledge engineer for methodological aspects and bootstrapping of the ontology, and, on the other hand, stakeholders for specific concepts and validation.

The terminological study was at the heart of ontology engineering, it provided the candidate terms for which consensual definitions have to be produced. These definitions expressed the intension of the concepts captured in the ontology. There were three cases:

- *One term* corresponding to one and only *one notion*: we labelled the notion with the term.
- *Several terms* corresponding to *one notion*: these terms were synonyms, we kept the list of synonyms and chose the most commonly used term to label that notion.
- *One term* corresponding to *several notions*: the term was kept, but noted as ambiguous and several expressions of intensions were defined with non ambiguous labels (e.g. compound terms).

The explicit representation of the two levels, namely terminological level (capturing labels) and intensional level (capturing notions), is a real need. Likewise tools assisting ontologists for the terminological aspects in ontology engineering or tools managing terminology to support users in their interactions with the system are needed.

The first step in building the ontology from collected textual material is the generation of informal lexicons. They are the first intermediary representation introducing the distinction between terminological level (terms used to denote the notions) and the intensional level (capturing the definitions of the notions).

7.4 Perspectives in populating and structuring the ontology

The obtained concepts are structured in a taxonomy. The principles behind this structure go back to Aristotle who defined a specie by giving its genus (genos) and its differentia (diaphora): the genus is the kind under which the species, the differentia characterises the species within that genus. Thus we started regrouping concepts firstly in an intuitive way, then iteratively organising and reviewing the structure following the extended Aristotelian principles given by Bachimont [2000]. These principles tend to eliminate multiple inheritance which is a problem with the role concepts making an extensive use of this mechanism. Indeed the ontology captured, for instance, the fact that end-users did not differentiate the roles from the persons playing them (e.g. a *person* is a *situable entity*, *i.e.,* a *person* can play the role of situated entity in a *situated* relationship). This leads to multiple inheritance to capture the different combination of roles an entity can play.

A possibility to avoid that would be to introduce multiple view points [Ribière, 1999] and limit the application of the Aristotelian principles to a point of view. An approach is proposed in [Kassel *et al.*, 2000] introducing semantic axis as means to group the types of criteria used for the differentia. The extended principles could then be applied to concepts inheriting for the same semantic axis. Likewise, the extensive work of Guarino and Welty [Guarino, 1992; Guarino and Welty, 2000] contributes to clean-up the theoretical foundations of ontology engineering: they provide definitions, theoretical framework and constraints to be satisfied by the taxonomy so that ontologists relying on these definitions can check some validity aspects of their subsumption links. The only problem is that, as far as we know, no tool is available to help an ontologist *do that work easily and independently of a formalisation language* (at the time of this writing, Guarino and Welty are testing the first tool in several projects); it is a titanic work to apply this theory to large ontologies. These contributions appeared to be adapted to validation of top ontologies ensuring, by extension, a minimal coherence in the rest of the ontology.

As I described in the state of the art, the three common approaches encountered in literature to build an ontology are:

- A *bottom-up approach*: the ontology is built by determining first the low taxonomic level concepts and by generalising them. This approach is prone to provide tailored and specific ontologies.
- A *top-down approach*: the ontology is built by determining first the top concepts and by specialising them. This approach is prone to the reuse of ontologies.
- A *middle-out approach*: core concepts are identified and then generalised and specialised to complete the ontology. This approach is prone to encourage emergence of thematic fields and to enhance modularity.

I first thought that it would be interesting to try a cross approach by merging bottom-up and top-down approaches: it would enable to associate the benefit of being specific with the ability to reuse other ontologies.

After experience, I am not convinced that there exists a purely top-down, bottom-up or middle-out approach. They seem to be the *three complementary perspectives of a complete methodology*. It seems to me that the activities of finding structure by specialisation from a generic concept, or by generalisation from a specific concept are concurrent processes present at every level of depth in the ontology (bottom, middle or top) and at different detail grains (concepts or groups of concepts).

The *holistic nature of knowledge seems to lead to the holistic nature of ontologies, and the holistic nature of ontologies leads to the holistic nature of methodologies to build them*. Of course, for a given case, an approach can mainly rely on one perspective (e.g. some ontologies of chemical substances made extensive use of bottom-up approach), but we would not oppose the different approaches: they rather represent three perspectives combined in ontology engineering. When

engineering an ontology, an ontologist should carry out the tasks defined in these three perspectives in parallel.

For instance, in our case:
- Top-down approach: we *studied existing top-ontologies*, and upper parts of relevant ontologies to structure our top part and recycle parts of existing taxonomies;
- Middle-out approach: we *studied different branches, domains, micro-theories* of existing ontologies as well as core subjects identified during data-collection. It helped us understand what were the main areas needed and regroup candidate terms;
- Bottom-up approach: we *used reports from scenario analysis and data-collection traces*, so as to list scenario specific concepts and then to regroup them by generalisation.

The different buds (top concepts, core concepts, specific concepts) opening out in the different perspectives were the origins of partial sub-taxonomies of O'CoMMA. The objective then is to ensure the joint of the different approaches and each time an event occurs in one perspective it triggers checks and tasks in the others.

For instance, if you discover in the bottom-up perspective that some specific concepts of a domain are relevant, it is interesting in the top-down perspective to try to find existing top level structure of this domain and in the middle-out perspective to find the central concept of this domain.

Populating and structuring the ontology require tasks to be performed in parallel in three perspectives: top-down, middle-out and bottom-up. The holistic nature of knowledge leads to the holistic nature of ontologies, and the holistic nature of ontologies leads to the holistic nature of methodologies to build them. Thus performing tasks in one perspectives triggers tasks and checks in the others.

7.5 From semi-informal to semi-formal

Starting from the informal lexicons I separated attributes and relations from the concepts and I obtained three tables (Stage 2 in Figure 46 page 215). These tables evolved from a semi-informal representation (tables of terms and synonyms and their definitions) towards semi-formal representation (taxonomic links, signatures of relations). Extracts from these tables are given as examples in Table 11, Table 12 and Table 13. They are intermediary and maturing working-documents bridging the gap from data collection to formalisation.

In the final version of this intermediate representation there were three tables:

- The table of *concepts* (see Table 11) giving: unique name of potential concepts (Class), the core concept they are close to or the thematic field they belong to (View), their inheritance links (Super Class), synonymous terms identified (Other Terms), a natural language definition of the notion behind the concepts to try to capture their intension (Natural Language Definition), the source in data collection they stem from (Provider).
- The table of *binary relations* (Table 12) giving: unique name of potential relations (Relation), the concepts they link (Domain and Range), the thematic fields they cross (View), their inheritance links (Super Relation), synonymous terms identified (Other Terms), a natural language definition of the notion behind the relation to try to capture their intension (Natural Language Definition), the source in data collection they stem from (Provider).
- The table of *attributes* (Table 13) giving: unique name of potential attributes (Attribute), the concept they are attached to (Domain), the basic type of the value taken by the attribute (Range Type), the thematic fields they belong to (View), their inheritance links (Super Relation), synonymous terms identified (Other Terms), a natural language definition of the notion behind the attributes to try to capture their intension (Natural Language Definition), the source in data collection they stem from (Provider).

The last column (Provider) introduces the principle of traceability of concepts, relations or attributes and it is interesting for the purpose of abstracting a methodology from the work done in CoMMA, to know what kind of contribution influences a given part of the ontology. It also enables to trace the effectiveness of reuse. When several sources are given, it means that the notion is a compromise between these different sources. This was a first attempt that is, of course, not sufficient and much more work is needed to capture and make explicit the rationale (this very Ph.D. is in a way a contribution to that point); this rationale proved to be important to enable end-users to appropriate themselves the ontology.

The structuring of an ontology consists of identifying the aspects which must be explicitly formalised for the scenarios and of refining the informal initial lexicons to augment their structure with the relevant formal dimensions against which the notion can be formally described.

Class	View	Super class	Other Terms	Natural Language Definition	Prov.
Thing	Top-Level			Whatever exists animate, inanimate or abstraction.	Us
Entity	Top-Level	Thing		Thing which exists apart from other Things, having its own independent existence and that can be involved in Events.	Us; Ph
...	...	...	...	...	...
Event	Top-Level; Event	Thing		Thing taking place, happening, occurring; usually recognized as important, significant or unusual	Us
Gathering	Event	Event		Event corresponding to the social act of a group of Persons assembling in one place	Us
Formal Gathering	Event	Gathering		Gathering agreeable to established mode, forms, conventions and requirements, methodical; well planned and organized, not incidental, sudden or irregular	Us
Meeting	Event	Formal Gathering		Gathering formally arranged for a particular purpose, usually in a dedicated room and/or around a table	Us
...	...	...	...	...	...
Person	Top-Level; Person;	Living Entity		Living Entity belonging to mankind, an individual human being.	Us
Professional	Organization; Person	Person		Person who does activities that are characteristic of some job/profession/occupation for a livelihood.	Us; Cy
Employee	Organization; Person	Professional		Professional who works for an organization in return for financial or other compensation. Disjoint with Self-employed Professional.	Us; Cy
...	...	...	...	...	...
Organizational Entity	Organization	Entity		Entity recognized by and within the organization	Us; EO
Organization Group	Organization	Organizational Entity		Organization Entity composed of other Organization Entity working together in a structured way for a shared purpose.	Us, To
Organization	Organization	Organization Group		Organization Group including both informal and legally constituted organizations.	Us
Organization Part	Organization	Organization Group		Organization Group which is a sub-organization of another Organization Group	Us
Group of Individuals	Organization	Organization Group		Organization Group composed of Organization Individual only.	Us
...	...	...	...	...	...
Document	Document	Entity		Entity including anything serving as a representation of thinking.	Us
Memo	Document	Document		Document corresponding to a message or other information in writing sent by one person or department to another in the same Organization.	Us
Newsgroup Message	Document	Document	Forum Message	Document corresponding to messages displayed on the Internet and devoted to the discussion of a specified topic	Us
...	...	...	...	...	...

Table 11 Extracts from original table of concepts

Relation	Domain	Range	View	Super Relation	Other Terms	Natural Language Definition	Sy	Tr	Re	Prov
Relation	Thing	Thing	Top-Level			Represent a connection between two things.				Us
Manage	Organizational Entity	Organizational Entity	Organization	Relation		Relation denoting that an Organizational Entity (Domain) is in charge/control of another .Organizational Entity (Range)				EO
Include	Organizational Group	Organizational Entity; Person	Organisation; Person	Relation		Relation denoting that an Organizational Entity (Domain) has as a part another Organizational Entity (Range).		Tr		Us
Employed by	Employee	Organization	Organization	Relation		Relation denoting that an Organization has an Employee working or doing a job for it and pays this Employee for it.				Us; EO
Has for Activity	Organizational Entity	Activity Field	Organisation; Activity	Relation		Relation denoting that an Organizational Entity is working in an Activity Field.				Us
Is Interested by	Organizational Entity	Interest Field	Organisation; Person	Relation		Relation denoting that an Organizational Entity is working in an Activity Field.				Us
...	...	...	...	...	...	...	...			

Table 12 Extract from the original Relations Table

Attribute	Domain	Range (Data Type)	View	Super Relation	Other Terms	Natural Language Definition	Prov
Family Name	Person	literal (string)	Person	Designation	Last Name; Surname	The name used to identify the members of a family	Us
Comments	Document	literal (string)	Document	Relation		Textual remark or observation about a document	Us
Designation	Thing	literal (string)	Top-Level	Relation		Identifying word or words by which a thing is called and classified or distinguished from others	Us
Creation Date	Document	literal (date & time)	Document	Relation		Date the document was created	Us
Beginning	Gathering	literal (date & time)	Event	Relation		Starting date of an gathering	Us
End	Gathering	literal (date & time)	Event	Relation		Ending date of an gathering	Us
Address	Location	literal (string)	Top-Level	Relation		Address of a location	Us
Phone Number	Location	literal (phone)	Top-Level	Relation		Phone number of a location	Us
Indication	Location	literal (string)	Top-Level	Relation		Textual signs/clues pointing to the location ("in the cupboard of the rest room")	Us

Table 13 Extract from the original Attributes Table

7.6 On a continuum between formal and informal

The *informal version of the ontology is not merely an intermediary step* that will disappear after formalisation; the formal form of an ontology must include the natural language definitions, comments, remarks, that will be used by humans trying to appropriate the ontology. "Ontologies have to be intelligible both to computers and humans" [Mizoguchi and Ikeda, 1997]. This plays an important role in documenting the ontology and therefore, in enabling reuse and maintenance of ontologies.

The tables previously described evolved from semi-informal to semi-formal until the taxonomic links were sufficiently explicit to be translated in RDFS by scripts (Stage 3 in Figure 46 page 215). The ontology content did not change, but its underlying structure evolved from informal tables to formal taxonomic relations usable by software. We call this translation time the formal toppling point: the point at which the ontology toppled over from informal structuring to formal structuring.

Figure 53 shows how RDFS can be used to implement the different levels introduced previously:
- the *terminological level* where collected terms are organised. Relations between the intensional level and the terminological level indicate possible labels for each intension (property rdfs:label). An intension with several terms linked to it (e.g. in Figure 53: C_4) is characteristic of synonyms. A term with several intensions linked to it (e.g. in Figure 53: T_2) is characteristic of the ambiguity of this term; this is a formal indicator of ambiguity that can be used in interfaces to detect when a disambiguation mechanism (e.g. dialog with the user to refine the meaning, request a choice between in a list of candidate notions) should be activated.
- the *intensional level* where the intensional structure of the ontology is formalised. Relations between the intensional and the extensional levels represent the instantiation of a concept. The bundles of relations link an intension to its extension (e.g. in Figure 53: C_8).
- the *extensional level* where the factual memory is organised (annotations, state of affairs, user profiles). An extension linked to several intensions (e.g. in Figure 53: C_6 and C_7) is characteristic of multi-instantiation.

We kept all the informal views using XSLT stylesheets (Stage 4 in Figure 46 page 215):
- the *initial terminological table* representing a kind of lexicon of the memory is recreated at any time by a stylesheet (seeFigure 54 - h.).
- the *tables of concepts and properties* that are intermediary representations I showed previously are recreated by two other stylesheets.
- *navigation and search between the conceptual and terminological levels* are achieved thanks to one stylesheet exploiting the label tag of the schema in order to search for concepts or relations linked to a term (Figure 54 - a and b).
- a new view as an indented *tree of concepts* with their attached definition as a popup window following the mouse pointer is constructed by one stylesheet (Figure 54 - g), but the process is heavy and the use of multiple inheritance makes the tree view too much redundant. Improvements for displaying lattices are needed here such as hyperbolic views, window clipping on the lattice, or other graphical widgets.
- *browsing in the taxonomy* of concepts and relations is enabled thanks to two stylesheets. As any of the stylesheets presented here they can handle different languages for the terminological level thus enabling us to switch for instance from English to French or, as one could imagine, from one jargon to another (Figure 54 - c, d and e; the XSLT code is shown in Figure 55). The user can also ask for the listing of the extension of a concept or a relation. Note that *a sample of this extension can play the role of examples* to ease understanding of a notion.
- from a concept, one can look for relations having a compatible signature (Figure 54 - f).

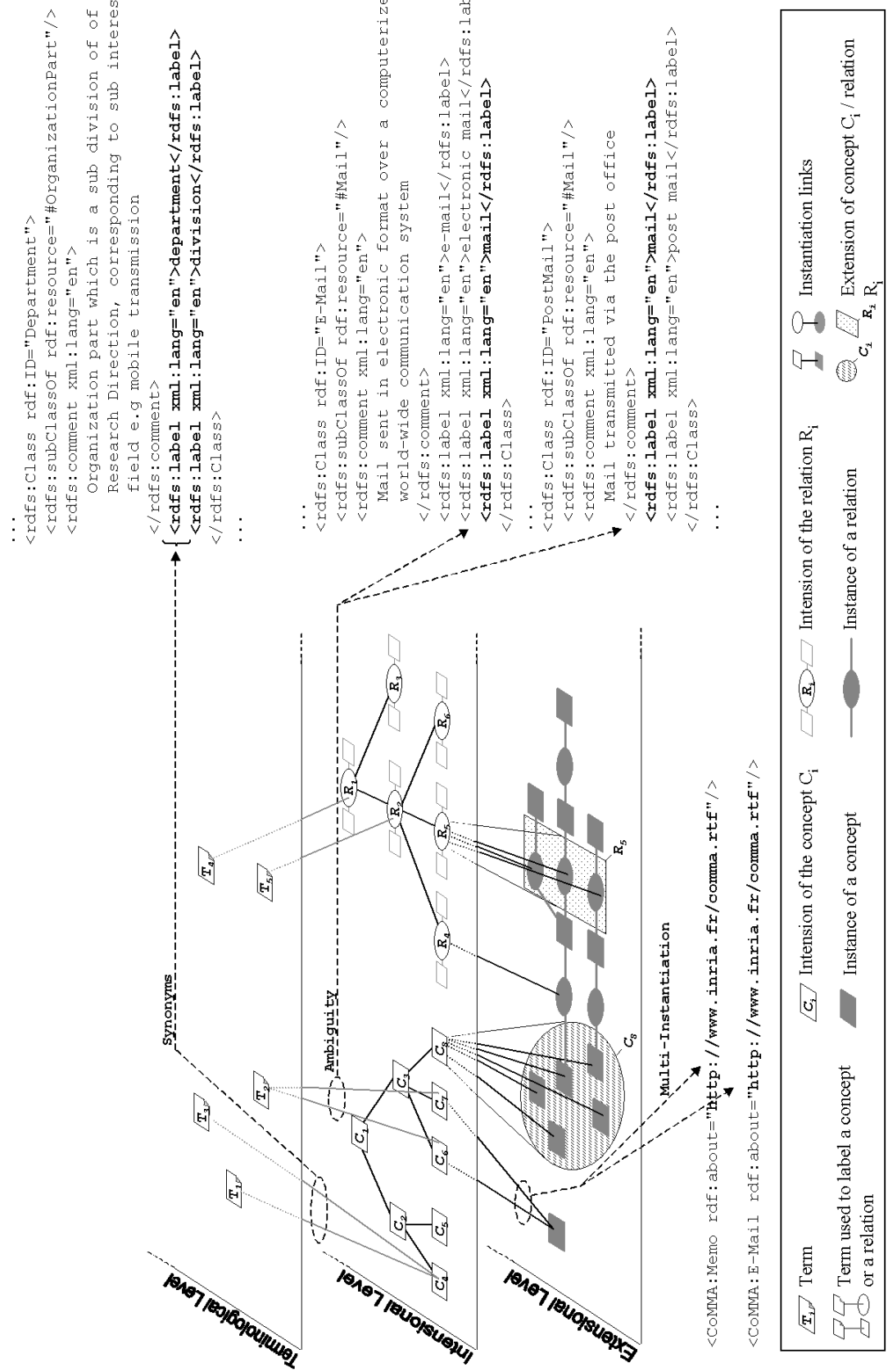


Figure 53 The terminological, intensional and extensional levels

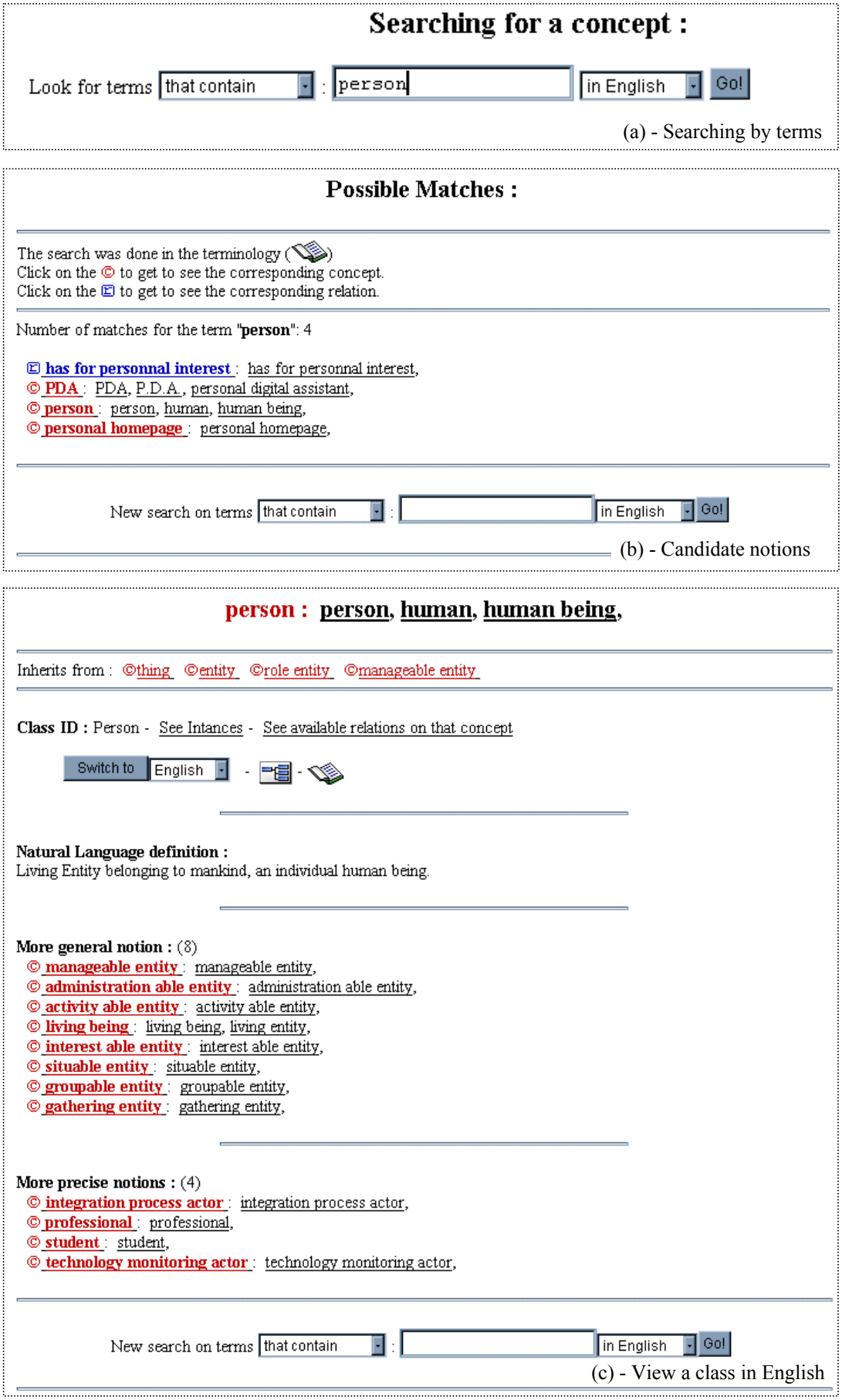


Figure 54 Browsing the ontology

personne : personne, humain, etre humain,

Inherits from : ©chose ©entite ©entite de role ©entite dirigeable

Class ID : Person - See Intances - See available relations on that concept

Switch to Francais -

Natural Language definition :
Entite vivante appartenant a l humanite, un etre humain individuel.

More general notion : (8)
© **entite dirigeable** : entite dirigeable,
© **entite capable d administrer** : entite capable d administrer,
© **entite capable d activite** : entite capable d activite,
© **etre vivant** : etre vivant, entite vivante,
© **entite capable d etre interessee** : entite capable d etre interessee,
© **entite localisable** : entite localisable,
© **entite groupable** : entite groupable,
© **entite de rassemblement** : entite de rassemblement,

More precise notions : (4)
© **acteur de la veille technologique** : acteur de la veille technologique,
© **acteur du processus d integration** : acteur du processus d integration,
© **etudiant** : etudiant,
© **professionnel** : professionnel,

New search on terms that contain : in English Go!

(d) - View a class in French

first name : first name, given name,

Inherits from : designation

Relation ID : FirstName - See Intances -

Switch to English -

[© **person** : person, human, human being,]--(first name, given name,)--[Text]

Natural Language definition :
The name that occurs first in a person s full name.

More general relation : (1)
designation : designation,

More precise relations :

New search on terms that contain : in English Go!

(e) - View a relation

Possible relations :

Click on the Ⓔ to get to see the corresponding relation.
Click on the © to get to see the corresponding concept.

[©person]--(Ⓔhas for ontological entrance point)--[©thing]

[©person]--(Ⓔcolleague)--[©person]

[©person]--(Ⓔfirst name)--[Text]

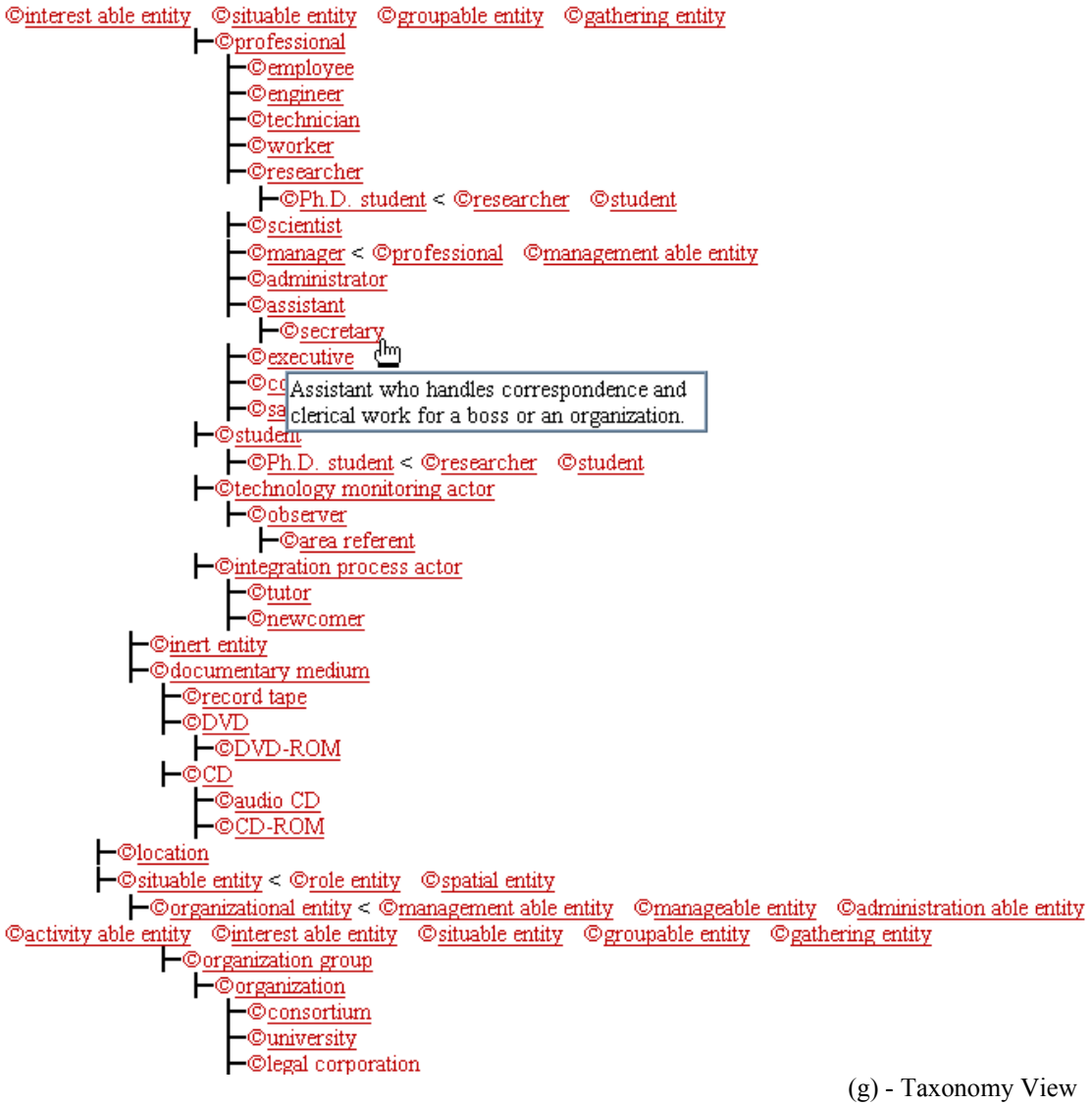
[©person]--(Ⓔname)--[Text]

[©person]--(Ⓔmobile number)--[Text]

[©person]--(Ⓔbirth date)--[Text]

[©management able entity]--(Ⓔmanage)--[©manageable entity]

(f) - Available relation for a concept



(g) - Taxonomy View

M.P.E.G. format	© MPEG	Data File Format of animated images/movie compressed in Moving Picture Experts Group format.
machine language	© machine language	The lowest-level Programming Language that consistw entirely of numbers.
machine learning	© symbolic learning	Domain interested in methods enabling the computers to learn.
magazine	© magazine	Document corresponding to a type of thin book with large pages which contains articles and photographs. It is usually intended to be a weekly or monthly paperback publication.
mail	© mail	Documents sent and delivered through a dedicated conveyance network.
mail	© e-mail	Mail sent in electronic format over a computerized world-wide communication system.
mail	© post mail	Mail transmitted via the post office.
make something	© make something	Activity in which something - tangible - is made from some raw materials.
male	© male	Person who belongs to the sex that cannot give birth.
manage	manage	Relation denoting that an entity is in charge/controls of another entity.
manageable entity	© manageable entity	Entity that can be managed.
management able entity	© management able entity	Entity that can manage another.
manager	© manager	Professional whose primary job is to manage other people, directing their work activity. A Manager tells his or her subordinate workers what to do.
manual	© manual	Reference Document which gives you practical instructions on how to do something or how to use something, such as a machine.
manufacture	© manufacture	Make Something from raw materials or component parts that are combined to produce a product.
map	© map	Document which, properly interpreted, models a region of physical space many times its own size by using

(h) - Terminology

The taxonomic tree view with its popup window is an interesting improvement. It is a first attempt to investigate how to proactively disambiguate navigation or querying. Before the user clicks on a concept, the system displays the natural language definition; this popup window invites the user to check his personal definition upon the definition used by the system and avoid misunderstandings.

These interfaces are "designed by an ontologist and for an ontologist" *i.e.,* I used them to browse, evaluate, build, show and discuss the ontology with the end-users. While these interface are not extremely useful for querying the memory, they enable users to familiarise themselves with the conceptualisation used by the system and to discuss about it with the ontologist. They also prove that XSLT is an interesting tool, modular enough to rapidly develop views adapted to a design or exploitation task.

Formalisation is an enriching process whereby the initial informal lexicons are augmented with additional logical structures formalising the aspects relevant for the application scenarios. Any intermediary representation should always be available as a view on the ontology even after the formalisation process is completed. Ontologies and their design must be made explicit to the humans and the systems that design, use and maintain them.

```
1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <!DOCTYPE xsl:stylesheet [<!ENTITY nbsp " ">]>
3   <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"
4    xmlns="http://www.w3.org/TR/REC-html40"
5    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
6    xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#" >
7
8   <xsl:output method="html" media-type="text/html" encoding="ISO-8859-1" omit-xml-declaration="no"/>
9
10  <xsl:param name='class' select="'Entity'"/>
11  <xsl:param name='language' select="'en'"/>
12
13  <xsl:variable name="signature_search"
    select='concat("/comma/servlet/search_signature/comma/comma.rdfs?language=",$language,"&amp;class=")'/>
14  <xsl:variable name="action_search"
    select='concat("/comma/servlet/keyword/comma/comma.rdfs?language=",$language,"&amp;term=")'/>
15  <xsl:variable name="new_search" select='"/comma/servlet/keyword/comma/comma.rdfs?"'/>
16  <xsl:variable name="class_browse"
    select='concat("/comma/servlet/class_browse/comma/comma.rdfs?language=",$language,"&amp;class=")'/>
17  <xsl:variable name="change_language" select='"/comma/servlet/class_browse/comma/comma.rdfs?"'/>
18  <xsl:variable name="tree_browse"
    select='concat("/comma/servlet/tree_concepts/comma/comma.rdfs?max=100&amp;language=",$language,"#")'/>
19  <xsl:variable name="terminology_browse"
    select='concat("/comma/servlet/terminology/comma/comma.rdfs?language=",$language)'/>
20  <xsl:param name="ns" select="'http://www.inria.fr/acacia/comma'"/>
21  <xsl:variable name="action_query" select="'/comma/servlet/corese?partial=true&amp;ns=xmlns:s='"/>
22  <xsl:variable name="query" select="'query'"/>
23  <xsl:variable name="concept_gif" select='"/comma/image/concept.gif"'/>
24  <xsl:variable name="tree_gif" select='"/comma/image/org_chart.gif"'/>
25  <xsl:variable name="terminology_gif" select='"/comma/image/book.gif"'/>
26
27  <!-- Match the root of the ontology -->
28  <xsl:template match="/">
29  <html>
30  <body LINK="#000000" VLINK="#000000" ALINK="#000000" BGCOLOR="#FFFFFF">
31  <xsl:apply-templates />
32  </body>
33  </html>
34  </xsl:template>
35
36  <!-- Match the beginning of the RDF structure -->
37  <xsl:template match="rdf:RDF"><xsl:apply-templates /></xsl:template>
38
39  <!-- Select the class to be displayed -->
40  <xsl:template match="rdfs:Class[@rdf:ID=$class]">
41  <!-- Show the terms in the right language (french / english) -->
42  <center><h2><FONT COLOR="#BB0000"><xsl:value-of
    select="rdfs:label[attribute::xml:lang=$language][position()=1]"/> :</FONT> <xsl:apply-
    templates select="rdfs:label[attribute::xml:lang=$language]"/> </h2></center>
43  <hr />
44  <!-- Show at least one inheritance path of the concept -->
45    Inherits from : 
46    <xsl:call-template name="inheritance">
47      <xsl:with-param name="child" select="substring-after(rdfs:subClassOf/@rdf:resource,'#')" />
48    </xsl:call-template>
49  <hr />
50  <br />
51  <b>Class ID : </b> <xsl:value-of select='@rdf:ID'/> - 
52  <A target="right" href="{$action_query}'{$ns}%23'&amp;{$query}=&lt;s:{$class}/&gt;">See
    Intances</A> - 
53  <!-- Form to change the language -->
54  <A target="right" href="{$signature_search}{$class}">See available relations on that concept</A>
55    <BLOCKQUOTE>
56      <FORM ACTION="{$change_language}">
57        <INPUT type="HIDDEN" name="class" value='{$class}'/>
58        <INPUT type="Submit" value="Switch to" />
59        <select name="language">
60        <option value='en'>English</option>
61        <option value='fr'>Francais</option>
62        </select>
63          -  <xsl:variable name="ID" select="@rdf:ID" /> <A
    HREF="{$tree_browse}{$ID}"><IMG ALIGN="TEXTTOP" BORDER="0" SRC="{$tree_gif}" /></A> - <A
    HREF="{$terminology_browse}"><IMG VSPACE="0" ALIGN="ABSBOTTOM" BORDER="0" SRC="{$terminology_gif}"
    /></A>
64      </FORM>
65    </BLOCKQUOTE>
66
```

```
67  <hr WIDTH="50%"/>
68  <br />
69  <!-- Show the definition with the right language (french / english) -->
70  <b>Natural Language definition :</b> <br />
71  <xsl:apply-templates select="rdfs:comment[attribute::xml:lang=$language]"/>
72  <br />
73  <hr WIDTH="50%"/>
74  <br />
75  <!-- Show more general notions -->
76  <b>More general notion :</b>
77  <xsl:for-each select="rdfs:subClassOf/@rdf:resource">
78    <xsl:variable name="motherID" select="."/>
79    <xsl:if test="position()=1"> (<xsl:value-of select="last()"/>)<br /></xsl:if>
80    <xsl:for-each select="/rdf:RDF/rdfs:Class[@rdf:ID=substring-after($motherID,'#')]">
81     <xsl:sort select="rdfs:label[attribute::xml:lang=$language]"/>
82     <xsl:call-template name ="concept"/>
83     <br />
84    </xsl:for-each>
85  </xsl:for-each>
86  <br />
87  <hr WIDTH="50%"/>
88  <br />
89  <!-- Show more precise notions -->
90  <b>More precise notions :</b>
91  <xsl:for-each select="/rdf:RDF/rdfs:Class[rdfs:subClassOf/@rdf:resource=concat('#',$class)]">
92   <xsl:sort select="rdfs:label[attribute::xml:lang=$language]"/>
93   <xsl:if test="position()=1"> (<xsl:value-of select="last()"/>)<br /></xsl:if>
94   <xsl:call-template name ="concept"/>
95   <br />
96  </xsl:for-each>
97  <br />
98  <hr />
99      <FORM ACTION='{$new_search}'>
100       <CENTER>New search on terms <select   name="comparator">
101       <option value='contains'>that contain</option>
102       <option value='egals'>that are equal to</option>
103       <option value='starts-with'>that begin with</option>
104       </select> :
105       <INPUT type='text' size='20' name='term' value=''/>
106       <select   name="language">
107       <option value='en'>in English</option>
108       <option value='fr'>en Francais</option>
109       </select>
110       <INPUT type="Submit" value="Go!" /></CENTER>
111     </FORM>
112 <hr />
113 </xsl:template>
114 <!-- Sub template to display a concept -->
115 <xsl:template name="concept">
116  <xsl:variable name="ID" select="@rdf:ID"/>
117    <A HREF="{$class_browse}{$ID}"><IMG BORDER="0" SRC="{$concept_gif}" /><b><FONT
    COLOR="#BB0000"><xsl:value-of
    select="rdfs:label[attribute::xml:lang=$language][position()=1]"/></FONT></b></A>: <xsl:apply-
    templates select="rdfs:label[attribute::xml:lang=$language]"/>
118 </xsl:template>
119 <!-- Sub template to display a term -->
120 <xsl:template match="rdfs:label[attribute::xml:lang=$language]"> <xsl:variable name="label_term"
    select="translate(.,' ','+')"/><A HREF="{$action_search}{$label_term}"><xsl:value-of
    select="."/></A>,</xsl:template>
121 <!-- Sub template to display a definition -->
122 <xsl:template match="rdfs:comment"><xsl:value-of select="."/><br /></xsl:template>
123 <!-- Sub template to calculate inheritance path (pb with multiple inheritance) -->
124 <xsl:template name = "inheritance" >
125 <xsl:param name="child"></xsl:param>
126 <xsl:for-each select="/rdf:RDF/rdfs:Class[@rdf:ID=$child]">
127   <xsl:call-template name="inheritance">
128     <xsl:with-param name="child" select="substring-after(rdfs:subClassOf/@rdf:resource,'#')" />
129   </xsl:call-template>
130   <A HREF="{$class_browse}{$child}"><IMG BORDER="0" SRC="{$concept_gif}" /><FONT
    COLOR="#BB0000"><xsl:value-of
    select="rdfs:label[attribute::xml:lang=$language][position()=1]"/></FONT></A> 
131 </xsl:for-each>
132 </xsl:template>
133 <xsl:template match="*" />
134 </xsl:stylesheet>
```

Figure 55 XSLT template for viewing a class

7.7 Axiomatisation and needed granularity

As expected, a limitation of RDFS appeared when formalising implicit information and background knowledge. For instance, when we declare that someone manages a group, it is implicit that this person is a manager. Thus the 'manager' concept should be a 'defined concept', *i.e.,* a concept having an explicit definition enabling this concept to be derived from other existing concepts whenever possible *i.e.*:

$$\texttt{manager(x)} \Leftrightarrow \texttt{person(x)} \wedge (\exists\ \texttt{y organisation(y)} \wedge \texttt{manage (x,y)})$$

However the notion of defined concept does not exist in RDFS, even though the ability to factorise knowledge in an ontology requires the ability to express formal definitions.

To overcome this limitation, the current version of the CoMMA system uses rules written in an RDF/XML language specially created for RDF(S) and CORESE to encode the logical implication existing between a pattern and a type *i.e.,* the side of the formal concept definition which gives the sufficient condition e.g.:

$$\texttt{person(x)} \wedge (\exists\ \texttt{y organisation(y)} \wedge \texttt{manage (x,y)}) \Rightarrow \texttt{manager(x)}$$

The other side of the implication *i.e.,* the necessary condition was not always used in CoMMA or only partially *i.e.,* existential facts are not derived:

$$\texttt{manager(x)} \Rightarrow \texttt{person(x)} \wedge (\exists\ \textbf{y organisation(y)} \wedge \texttt{manage (x,y)})$$

Indeed this side of the definition leads to complete the base with existential facts (like the one in bold in the previous example) which were not found useful in our application scenarios. This choice is particular to CoMMA and in another context we may have used both sides. The very same needs apply to properties and, for instance, the *co-author* property could be defined by:

$$\texttt{co-author(x,y)} \Leftrightarrow \texttt{author(x,y)} \wedge (\exists\ \texttt{z person(z)} \wedge \texttt{author(z,y)} \wedge \neg(\texttt{z}\equiv\texttt{x}))$$

Axiomatic knowledge includes formal definition of notions, algebraic properties of relations, and additional axioms of the domain. It is crucial for adding intelligence to the semantic manipulations of the system. In a corporate semantic Web context, the needs for inference rules is clear: they are the key to discover knowledge that may be only implicit because of the point of view adopted when annotating and thus improve the information management mechanisms. Rules are viewed as the explicit factorisation of knowledge that may be implicit in some annotations.

Another aspect addressed was the algebraic properties of relations. The RDFS mark-up was extended to qualify:

- the symmetry: for instance, the property *related to* is symmetric *i.e.,* "if resource R_1 is *related to* a resource R_2" then "resource R_2 is *related to* a resource R_1"
- the transitivity: for instance *include* is transitive *i.e.,* "if cluster *C includes* division *D*" and "division *D includes* team *T*" then "cluster *C includes* team *T*"
- the reflexivity: for instance the relation is *acquainted to* is reflexive *i.e.,* "any person *P* is *acquainted to P*" (anyone is acquainted to oneself).
- the inverse: for instance *member of* is the inverse of *include i.e.,* "if person *P* is *member of* a group *G*" then "group *G includes* a person *P*"

Axiomatic knowledge increases the granularity of an ontology.

The notion of *colleague*, for instance, summarises very well the whole problem. The term was collected during analysis of technology monitoring scenario where "an observer may want to send an annotation to some persons and their colleagues or look for documents written by some persons or their colleagues". The term was defined and first modelled as a concept. It turned out that no one is intrinsically a *colleague* by oneself; thus I modelled it as a relation "*x* is a colleague of *y*". This relation needed to be typed as symmetric: it needs the extension for algebraic properties of relations. Finally, considering the application scenarios, it appeared that this relation would not be used for annotation, but that it would more likely be inferred from the description of the state of affairs of the organisational structure e.g. "Mr. *X* and Ms. *Y* belong to department *D* therefore, there is a 'colleague' relation between *X* and *Y*". Thus we needed the ability to formally define the sufficient conditions that make two people *colleagues*; therefore, we introduced rules for such inferences.

In the current version of the system, the formal definition of *colleague* was coded in a rule as shown in Figure 56. This rule is treated as a rule graph in conceptual graphs by an inference engine working in forward-chaining. The engine exploits these rules to complete the annotation base with deducible implicit facts. Figure 57 shows some examples of rules viewed through adequate XSLT templates and Figure 58 shows the result of a query using the *colleague* property, while no instance of this property was explicitly made in an annotation.

Axiomatisation provides the ability to factorise knowledge in the ontology and detect implicit knowledge in facts that may have been described from a particular point of view. The logical expressiveness determines the possible granularity of the formal aspects of the ontology. The expressiveness has to be chosen so as to allow the granularity required by the application scenarios. Notions are knowledge enabling to define other knowledge, but to be defined these notions themselves require additional notions; an ontology structure could appear as a fractal structure where one could zoom in the conceptual structure, looking at finer granularity and wandering in the definitional loops that form knowledge.

```
1   <cos:rule>
2       <cos:if>
3         <rdf:RDF>
4             <CoMMA:OrganizationalEntity>
5               <CoMMA:Include> <CoMMA:Person rdf:about="?x"/> </CoMMA:Include>
6               <CoMMA:Include> <CoMMA:Person rdf:about="?y"/> </CoMMA:Include>
7             </CoMMA:OrganizationalEntity>
8         </rdf:RDF>
9       </cos:if>
10
11      <cos:then>
12        <rdf:RDF>
13            <CoMMA:Person rdf:about="?x">
14              <CoMMA:Colleague> <CoMMA:Person rdf:about="?y"/></CoMMA:Colleague>
15            </CoMMA:Person>
16        </rdf:RDF>
17      </cos:then>
18  </cos:rule>
```

Figure 56 Rule defining Colleague relation

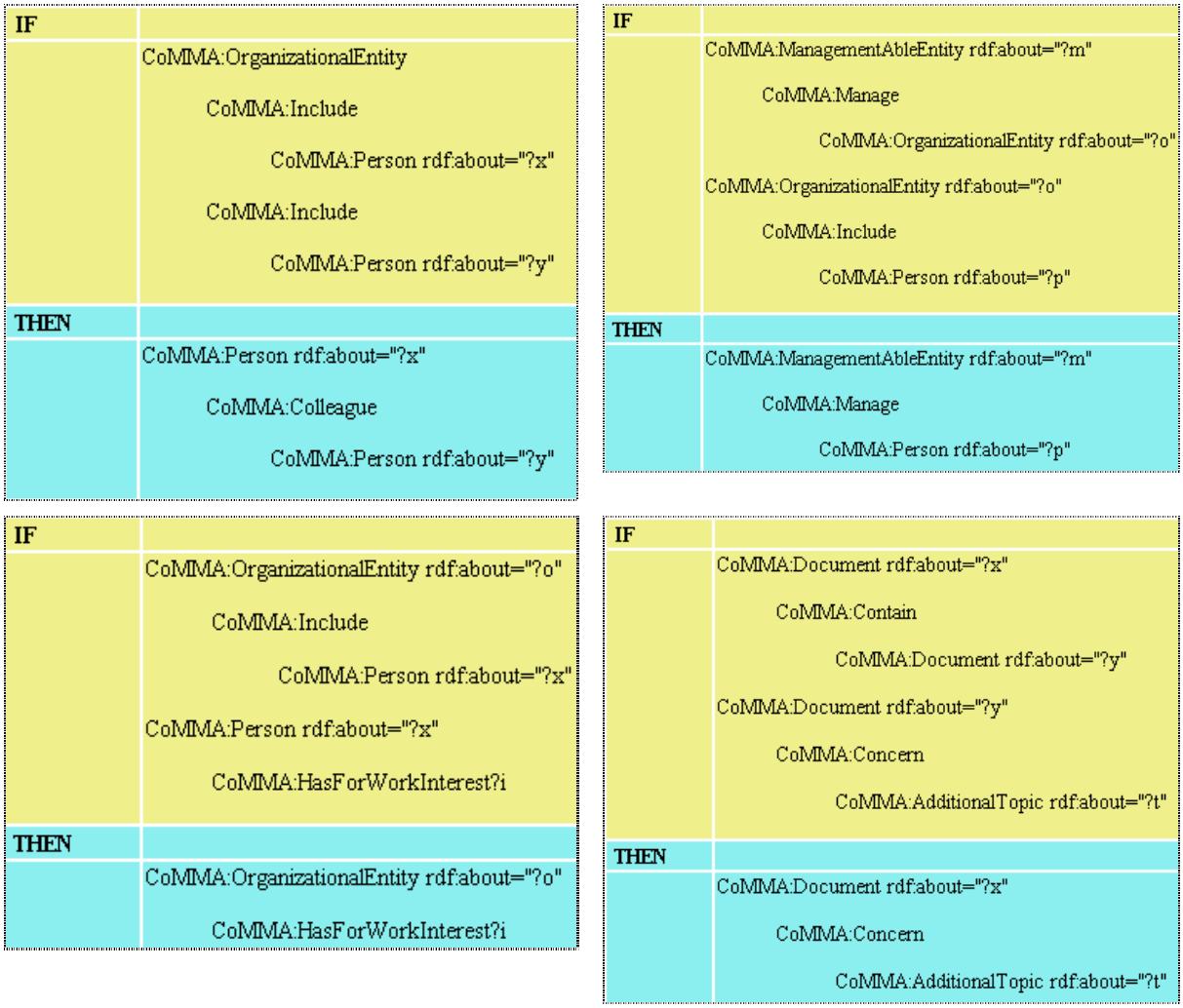


Figure 57 Some rules of CoMMA displayed through XSLT stylesheets

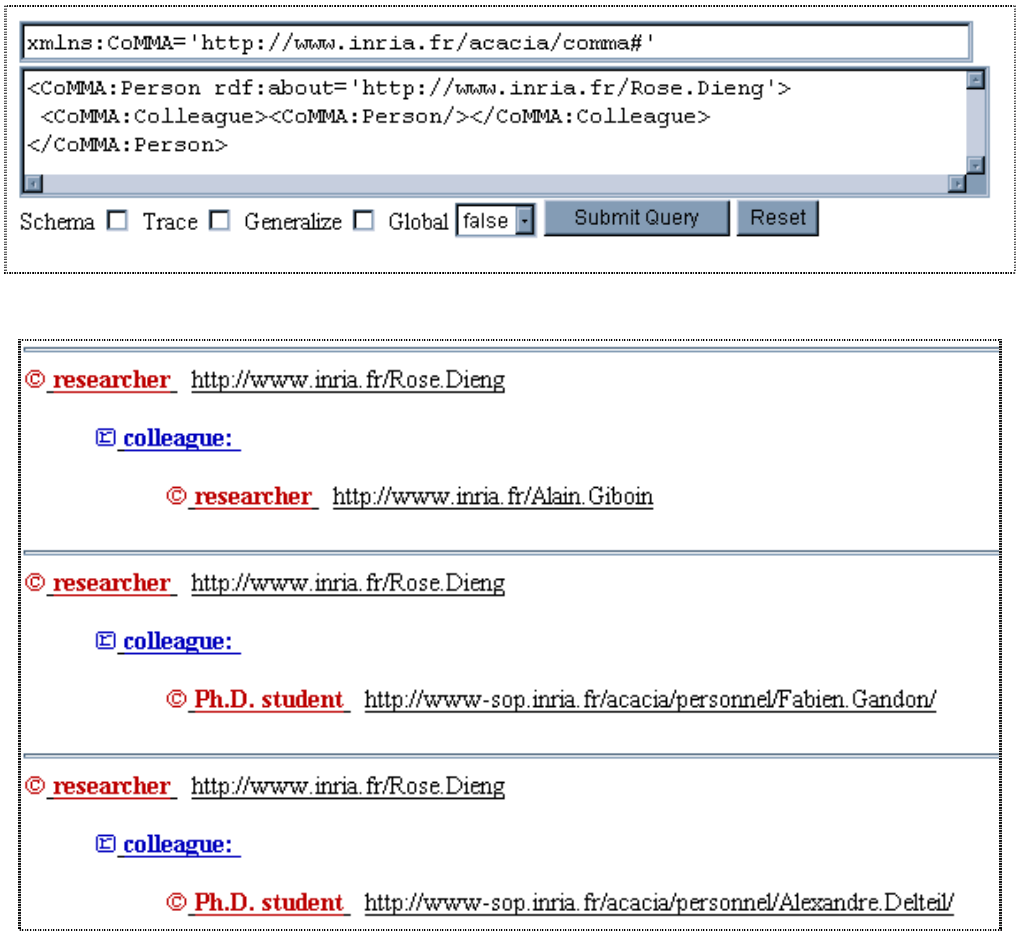


Figure 58 Querying the base with the defined relation 'Colleague'

7.8 Conclusion and abstraction

To summarise the approach taken to build O'CoMMA and abstract some conclusions, I would like to recall the highlighted methodological points and identify main algorithmic steps in a pseudo-code informal description given in Figure 59:

```
1   //------------------------------------------------------------------------ Initialisation ---
2   Identify and capture application scenarios in informal scenario reports
3   Initialise set of terms T_I := {terms appearing in scenarios reports}
4   //------------------------------------------------------------------------------ Collection ---
5   For each term T ∈ T_I
6    {
7     Choose and apply realistic data collection methods to capture information about T
8     Identify and capture possible existing ontology or external resources concerning T
9     Identify T_R := { terms T' related to T }
10    For each term T' ∈ T_R
11      If (T' ∉ T_I) and (T' mobilised by usage in scenarios)
12      Then T_I := T_I ∪ {T'}
13   }
14  //------------------------------------------------------- Lexicon completion and reuse ---
15  Initialise lexicon L_x := ∅
16  For each term T ∈ T_I
17   {
18     Identify set of definitions D_T := {definitions attached to T in different use contexts}
19     D_T := D_T ∪ { existing definitions of T found in other ontologies or resources }
20     For each definition D ∈ D_T
21       If D is relevant for usage in scenarios
22        {
23         If ∀ notion N'∈L_x with a definition D' we have D ≠ D'
24          {
25           Identify set of synonyms S_T := { synonyms of T for definition D}
26           Choose unique label L for denoting D
27           Initialise notion N := (L,S_T,D)
28           L_x := L_x ∪ { N }
29          }
30         Else
31          {
32           Select notion N' with a definition D' such that D = D'
33           Select S_T' set of synonyms of N'
34           S_T' := S_T' ∪ T
35          }
36        }
37  //------------------------------------------------------------------------- Formalising ---
38  Initialise set of formal aspects A_F := ∅
39  For each notion N ∈ lexicon L_x
40   {
41    Identify set A_D := {definitional aspects of N to be formalised for usage in scenarios}
42    A_F := A_F ∪ A_D
43   }
44  For each kind of expressiveness K_E required by A ∈ A_F
45    Refine structure of lexicon L_x to enable capture of K_E
46    // e.g.: concept vs. relation implied a separation, subsumption implied a column parent,
47    //       signature for relations, etc.
48  For each formal aspect A ∈ A_F
49    Populate structure of lexicon L_x for each notion N for which A is relevant
50    // e.g.: person is a concept, colleague a relation, etc.
51  Choose formal language L_F such that expressiveness(L_F) > expressiveness(A_F)
52  Formal ontology O_F := translation(L_x, L_F)
53  //------------------------------------------------ Validation of coverage and lifecycle ---
54  While (O_F is used)
55    If a lack l is detected in O_F then
56     {
57      If l is due to a set of missing terms T_M
58      Then T_I := T_I ∪ T_M and loop to step(5)
59      If l is due to a set of missing formal aspects A_M
60      Then A_F := A_F ∪ A_M and loop to step(44)
61     }
```

Figure 59 Schematic overview of the ontology engineering process

This description can be commented as follows:

- The work starts with an *inventory of fixture*; the end-users are invited to describe two types of scenarios: *current scenarios* where needs were detected and *ideal scenarios* they would like to achieve. Scenarios grids and reports are used to initialise, focus and evaluate the whole knowledge modelling and the whole application design. Concerning the ontology, informal scenario reports provide the initial base for terminological study of the application context.
- This first set of terms initialise the *data collection and analysis* activities such as interviews, observation, document analysis, brainstorming, brainwriting and questionnaires. It enables the designers to discover implicit and hidden aspects of the conceptualisation(s) underlying the scenarios and to be in contact with real cases. Since collection is time-consuming and resource-consuming *scenario-based analysis* is used to guide and focus the whole collection, analysis and design processes. The involvement and training of end-users in these tasks is highly interesting to avoid access restriction problems, to improve the capture of user's conceptualisation and to prepare the ontology future maintenance.
- From the collected material, *lexicons are built to capture the terms and definitions* mobilised by the scenarios. Lexicons constitute the first intermediary representation towards the final ontology and they introduce the separation and links between terms used to denote the notions and the definitions of the notions. To build these lexicons, terms are analysed in context and reuse of existing ontologies or resources providing candidate definitions, is pursued wherever possible, using semi-automatic tools to scale up the process when they are available. As rightly noticed by Joost Breuker these ontologies should be considered as sources of inspirations; indeed these ontologies were not reused *per se*, but their conceptualisation choices and their definitions inspired my design choices on many occasions.
- Organisations are parts of broader organisations, cultures, etc. So *external relevant resources* may have to be included in the collection and analysis processes; knowledge is holistic and its collection is bounded by the scope of the scenarios, and not by the organisational boundaries. Built lexicons contain one and only one instance of each definition, with an associated label and a set of synonymous terms that may be used to denote it.
- The *structuring of an ontology* consists in identifying the aspects which must be explicitly formalised for the scenarios and in refining the informal initial lexicons to augment their structure with the relevant formal dimensions against which the notions can be formally described. Formalisation is an enriching process whereby the initial informal lexicons are augmented with additional logical structures formalising the aspects relevant for the application scenarios. Ontologies and their design must be kept explicit to both the humans and the systems that design, use and maintain them.
- Both populating and structuring the ontology require tasks to be performed in parallel in *three complementary perspectives*: top-down (by determining first the top concepts and by specialising them), middle-out (by determining core concepts and generalising and specialising them) and bottom-up (by determining first the low taxonomic level concepts and by generalising them). Thus the holistic nature of knowledge leads to the holistic nature of ontologies, and the holistic nature of ontologies leads to the holistic nature of methodologies to build them in an event-driven fashion: performing a task in one perspective triggers tasks and checks in the other perspectives.
- Coverage of the ontology is evaluated in terms of *exhaustivity, specificity, and granularity* against usage scenarios. Any missing coverage triggers additional collection and/or formalisation. Granularity improvements require axiomatisation to factorise knowledge in the ontology and detect implicit knowledge in facts that may have been described from a particular point of view.

Ontologies being living objects, over time their uses will reveal new needs for knowledge acquisition and formalisation following a never-ending prototype lifecycle.

8 The ontology O'CoMMA

The ontology O'CoMMA is the result produced by the method presented in the previous chapter. In this part, I discuss the characteristics of O'CoMMA and its structure, and I try to objectively evaluate some aspects of it.

First, I describe the overall structure of O'CoMMA and the interest of a division into layers and domain. Then, I describe each part, the vocabulary they provide and its use: the abstract top layer of O'CoMMA, the part dedicated to the organisational structure modelling, the part dedicated to the description of people, the part dealing with domain topics, and the properties linking these concepts. Finally, I shall discuss some extensions of O'CoMMA and appropriation experiences with end-users.

8.1 Overall structure of O'CoMMA

O'CoMMA contains: 470 concepts organised in a taxonomy with a depth of 13 subsumption links; 79 relations organised in a taxonomy with a depth of 2 subsumption links; 715 terms in English and 699 in French to label these primitives; 547 definitions in French and 550 in English to explain the meaning of these notions.

O'CoMMA is divided into three main layers (see Figure 60):

- A very general top that looks like other top-ontologies. More pricesely, the top part is not a complete upper ontology but rather the relevant sub-part of such an upper ontology on which the other layers can rely, to meet the needs for the considered scenarios;
- A very large and ever growing middle layer divided in two main branches: (1) one generic to corporate memory domain (documents, organisation, people...) and (2) one dedicated to the topics of the application domain (telecom: wireless technologies, network technologies...);
- An extension layer which tends to be scenario-specific and company-specific with internal complex concepts (Trend analysis report, Area referent, New Employee Route Card...).

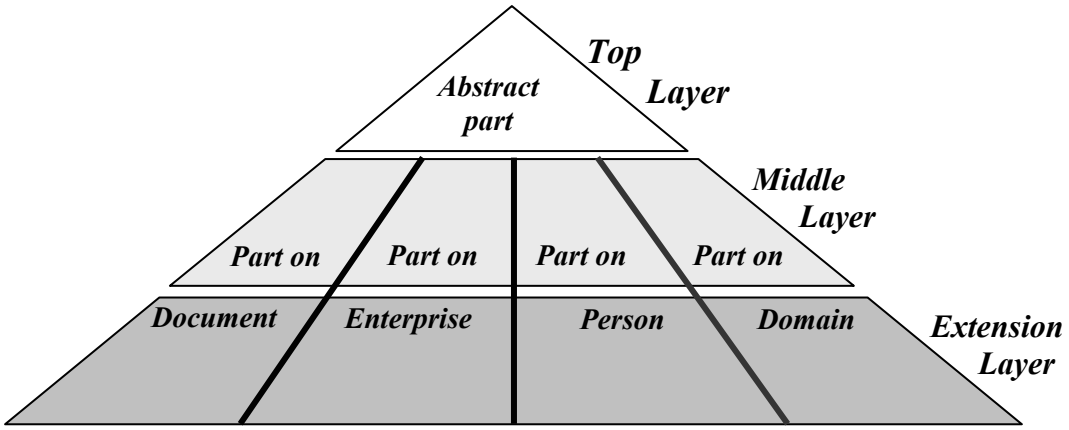


Figure 60 Architecture of O'CoMMA

The equilibrium between usability and reusability of notions of the ontology varies within the ontology:

- The upper part is extremely abstract and the first part of the second layer describes concepts common to corporate memory (e.g. person, employee, document, report, group, department, ...), therefore, they both seem to be reusable in other application scenarios.
- The second part of the middle-layer deals with the application domain (in our case telecom and building industry). Therefore, it would be reusable for scenarios only in the same application domain.
- The last layer extends the two previous parts with specific concepts that should not be reusable as soon as the organisation, the scenario or the application domain change.

On a conceptual plan, an ontology can be seen as a fractal *i.e.,* a conceptual structure composed of similar conceptual structures; however, the usability and reusability characteristics are not homogeneous over this structure and they tend to vary in a conversely proportional equilibrium, going from more reusable top notions to more usable bottom notions.

8.2 Top of O'CoMMA

To unify all the branches of O'CoMMA in a tree, and enable generalisation, the top of the ontology defines a set of concepts extended by the other parts:

CONCEPTS	DEFINITION
thing	Whatever exists animate, inanimate or abstraction.
entity	Thing which exists apart from other Things, having its own independent existence and that can be involved in Events.
additional topic	Entity representing subjects that hold attention and possibly something one wants to discover. These topics are additional in the sense that they were not introduced or used anywhere else in the ontology, but were identified as relevant for document annotation - domain concepts, general subjects...- (*)
...	
document	Entity including elements serving as a representation of thinking.
...	
non spatial entity	Entity that has no spatial nature and does not pertain to space.
activity, attribute, pattern, consultation trace, push mode trace, ...	
role entity	Entity that can play a role in a relation.
...	
spatial entity	Entity pertaining to or having the nature of space.
...	
time entity	Entity related to the continuum of experience in which events pass from the future through the present to the past.
...	
event	Thing taking place, happening, occurring and usually recognised as important, significant or unusual.
corporate memory event	Event corresponding to changes in the Corporate Memory (**)
addition	Corporate Memory Event corresponding to content added to the memory.
consultation	Corporate Memory Event corresponding to a user seeking information from the memory
deletion	Corporate Memory Event corresponding to content removed from the memory.
modification	Corporate Memory Event corresponding to content transformed in the memory
update	Modification corresponding to content changed into more recent one.
entertainment event	Event occurring primarily to amuse or entertain Persons.
...	
gathering	Event corresponding to the social act of a group of Persons assembling in one place.
...	

(*) Introduced as the top of the domain branch
(**) Introduced to enable extensions of the agent ACL to describe the system itself

Table 14 Top of O'CoMMA

8.3 Ontology parts dedicated to organisational modelling

In both scenarios, one of the most relevant aspects of the organisational state of affairs is the company structure. An overview of the main ontology primitives needed for the description at play in our scenarios is presented in Table 15:

CONCEPTS	DEFINITION
organisational entity	Entity recognised by and within the organisation.
organisation group	Organisational entity which is composed of other Organisational Entities working together in a structured way for a shared purpose.
group of individuals	Organisation Group composed of individuals only.
association, club, project, union, unit (*)	
international organisation group	Organisation Group of international scope, that is, one which has substantial operations, physical facilities, or substantial membership in multiple countries.
local organisation group	Organisation Group of local scope, that is, members distributed in a local area - a Neighbourhood, City, rural region, etc.- or having a local area of activity and concern.
national organisation group	Organisation Group of nation-wide scope, that is distribution throughout some Country of its members and/or activities.
organisation	Organisation Group including both informal and legally constituted organisations.
consortium, legal corporation, university (*)	
organisation part	Organisation Group which is a sub-organisation of another Organisation Group.
cluster, department, division, direction, research direction (*)	
single site organisation	Organisation Group which has a single location as its physical quarters.
* *(see ontology for more details)*	

Table 15 An overview of the concepts used for organisational modelling

This table extracted from O'CoMMA also shows that it may be difficult to respect all theoretical principles. For instance the necessity of having one differentia under a father as implicitly advocated by extended Aristotelian principles is not respected here. On the other hand multi-inheritance has been avoided as much as possible within the view (we shall see however that ability to play roles in relations introduced lots of multi-inheritance to merge all the abilities an object may have).

To describe and link these concepts, some relations are proposed see for instance Table 16 (more about relations is said in section 8.7).

RELATION	DEFINITION
administer	Relation denoting that an entity regulates the operations of an organisational entity.
include	Relation denoting that an organisational entity has another organisational entity as a part.
employed by	Relation denoting that an employee works in an organisation and that this organisation pays him or her.
manage	Relation denoting that an entity is in charge of /controls another entity.
has for activity	Relation denoting that an entity carryies out an activity.
is interested in	Relation denoting that an entity is interested in a topic.
situated	Relation denoting that an entity is located in a location.
see ontology for more details (signature, inheritance,...)	

Table 16 Extract of the relations used in organisational modelling

These concepts and relations can then be used to describe the enterprise model and capture some relevant aspects such as the departments, their employees and their interest, etc. Figure 61 shows an example of what has been modelled at Deutsch Telekom by an end-user from the T-Nova department for a trial in CoMMA.

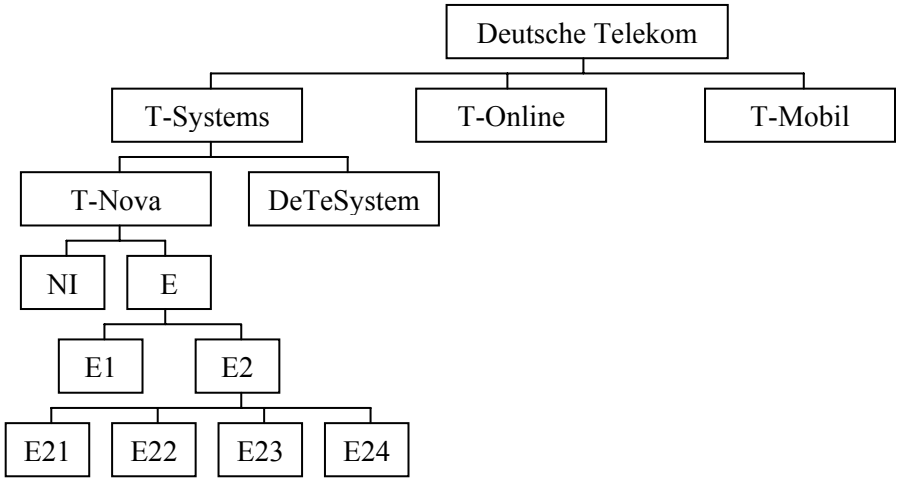


Figure 61 Example of T-Nova organisation

Links in Figure 61 denote aggregation (e.g. T-Systems includes T-Nova and DeTeSystem). Label such as 'NI', 'E22', etc. denote internal acronyms for sub-divisions.

Other organisational aspects are described likewise such as employees (technician, student, researcher, etc.), domain topics (building materials, telecom), documents (report, news, etc.). For the sake of presentation, I shall not detail the entire ontology here and only briefly comment these parts. However each part will be used to show some different aspects of modelling. I invite the reader to consult the appendix for more details.

8.4 Ontology parts dedicated to user profiles

Profiles are a special type of document, describing some aspects of individual users or of groups of users that have been judged relevant for the system inferences.

CONCEPTS	DEFINITION
Profile	Document containing a biographical sketch.
Individual Profile	Profile concerning one individual.
Group Profile	Profile concerning a group of people.
see ontology for more details	

Table 17 Types of profiles

Just like the organisational model, a profile is also an annotation that uses a number of ontological primitives to describe the user or the group. Figure 62 is an example of information that can be given about a user.

This profiles says it is itself a profile (line 1 to 3). Then, from line 5 to 45 it gives information about the engineer facet of Fabien GANDON, his name, first name, birth date, work interest and personal interests. Additionally the profile gives "ontological entrance points" *i.e.,* concepts the user is familiar with and that the system can use as entrance point when an interface requires to browse the ontology for instance to build a query. These entrance points can be derived from interests, activities or from results of machine learning techniques applied to the study of previous usage of the system by the user. Further explanations on that point are given later. From line 45 to 63 the employee facet of Fabien GANDON is described with his hiring date, the type of contract, and the activities he performs. The profile is presented here as being in one file, but in fact, the employee facet was in a separate annotation of the organisational model.

An interesting point to note here is the reification of some classes as objects. For instance, line 54 to 56 say that Fabien GANDON has for activity research and reifies the class 'research' by creating an instance of 'research' having for URI the one of the class 'research':

```
<CoMMA:Research rdf:about="http://www.inria.fr/acacia/comma#Research"/>
```

This is used to create a global object representing the "research activity". It allows to say that if another person has for activity 'research', say Rose DIENG, then using the same convention both objects will have the same URI, therefore, they will be treated as one object thus capturing the following meaning: when two persons are carrying an activity of research, it is the same activity. This does not mean that the research results, methods, etc. are the same, but that the activity is the same.

Reification of classes as objects of their own type having for URI the one of the class is a possible convention to create global objects with universal identifier.

Relations such as 'is interested in' are also exploited for pushing documents; this will be explained in a later section.

```
1   <CoMMA:IndividualProfile rdf:about="#">
2     <CoMMA:Title>Employee profile of Fabien GANDON</CoMMA:Title>
3   </CoMMA:IndividualProfile>
4
5   <CoMMA:Engineer rdf:about="http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/">
6     <CoMMA:FamilyName>GANDON</CoMMA:FamilyName>
7     <CoMMA:FirstName>Fabien</CoMMA:FirstName>
8     <CoMMA:BirthDate>31-07-1975</CoMMA:BirthDate>
9
10    <CoMMA:HasForWorkInterest>
11      <CoMMA:MultiAgentSystemTopic
            rdf:about="http://www.inria.fr/acacia/comma#MultiAgentSystemTopic"/>
12    </CoMMA: HasForWorkInterest >
13    <CoMMA:HasForWorkInterest>
14      <CoMMA:KnowledgeEngineeringTopic
            rdf:about="http://www.inria.fr/acacia/comma#KnowledgeEngineeringTopic"/>
15    </CoMMA:HasForWorkInterest>
16    <CoMMA:HasForWorkInterest>
17      <CoMMA:JavaProgrammingTopic
            rdf:about="http://www.inria.fr/acacia/comma#JavaProgrammingTopic"/>
18    </CoMMA:HasForWorkInterest>
19    <CoMMA:HasForWorkInterest>
20      <CoMMA:XMLTopic
            rdf:about="http://www.inria.fr/acacia/comma#XMLTopic"/>
21    </CoMMA:HasForWorkInterest>
22
23    <CoMMA:HasForPersonalInterest>
24      <CoMMA:MusicTopic rdf:about="http://www.inria.fr/acacia/comma#MusicTopic"/>
25    </CoMMA:HasForPersonalInterest>
26    <CoMMA:HasForPersonalInterest>
27       <CoMMA:HumanScienceTopic
                rdf:about="http://www.inria.fr/acacia/comma#HumanScienceTopic"/>
28    </CoMMA:HasForPersonalInterest>
29
30    <CoMMA:HasForOntologicalEntrancePoint>
31      <CoMMA:ComputerScienceTopic
                rdf:about="http://www.inria.fr/acacia/comma#ComputerScienceTopic"/>
32    </CoMMA:HasForOntologicalEntrancePoint>
33    <CoMMA:HasForOntologicalEntrancePoint>
34      <CoMMA:Service rdf:about="http://www.inria.fr/acacia/comma#Service"/>
35    </CoMMA:HasForOntologicalEntrancePoint>
36    <CoMMA:HasForOntologicalEntrancePoint>
37      <CoMMA:Document rdf:about="http://www.inria.fr/acacia/comma#Document"/>
38    </CoMMA:HasForOntologicalEntrancePoint>


39    <CoMMA:HasForOntologicalEntrancePoint>
40      <CoMMA:Person rdf:about="http://www.inria.fr/acacia/comma#Person"/>
41    </CoMMA:HasForOntologicalEntrancePoint>
42    <CoMMA:HasForOntologicalEntrancePoint>
43    <CoMMA:OrganizationGroup
          rdf:about="http://www.inria.fr/acacia/comma#OrganizationGroup"/>
44    </CoMMA:HasForOntologicalEntrancePoint>
45   </CoMMA:Engineer>
46
47   <CoMMA:Employee rdf:about="http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/">
48    <CoMMA:HireDate>1999-11-02</CoMMA:HireDate>
49    <CoMMA:EmployedBy>
50       <CoMMA:LocalOrganizationGroup rdf:about="http://www.ac-nice.fr/" />
51    </CoMMA:EmployedBy>
52    <CoMMA:EmploymentContract> <CoMMA:Temporary/> </CoMMA:EmploymentContract>
53
54    <CoMMA:HasForActivity>
55      <CoMMA:Research rdf:about="http://www.inria.fr/acacia/comma#Research"/>
56    </CoMMA:HasForActivity>
57    <CoMMA:HasForActivity>
58      <CoMMA:Development rdf:about="http://www.inria.fr/acacia/comma#Development"/>
59    </CoMMA:HasForActivity>
60    <CoMMA:HasForActivity>
61      <CoMMA:Education rdf:about="http://www.inria.fr/acacia/comma#Education"/>
62    </CoMMA:HasForActivity>
63   </CoMMA:Employee>
```

Figure 62 Example of User description

The profile then uses other primitives to describe the history of use. It was a requirement for applying machine learning techniques for adaptation to user. An important concept here is the consultation trace. An example of such a trace is given in Figure 63:

```
1   <CoMMA:ConsultationTrace>
2    <CoMMA:Visitor>
3     <CoMMA:Employee rdf:about= "#http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/"/>
4    </CoMMA:Visitor>
5    <CoMMA:VisitedDocument>
6     <CoMMA:Memo
          rdf:about= "http://www.myintranet.com/Ada.Lovelace/projects/TIGRA/note28-10.doc"/>
7    </CoMMA:VisitedDocument>
8    <CoMMA:FirstVisit>2000-11-21</CoMMA:FirstVisit>
9    <CoMMA:LastVisit>2000-12-07</CoMMA:LastVisit>
10   <CoMMA:VisitCount>17</CoMMA:VisitCount>
11   <CoMMA:RatingGiven><CoMMA:GoodRating/></CoMMA:RatingGiven>
12  </CoMMA:ConsultationTrace>
```

Figure 63 Example of Consultation Trace

This trace says that the employee Fabien GANDON, consulted a memo at a given URL. He consulted it 17 times, the first time was in 2000-11-21 and the last one was 2000-12-07 (US date format). He appreciates this document (good rating). This information is used by the machine learning algorithm together with the annotations about the memo to try to derive a partial order relation over the topics proposed by the ontology, and to use this relation for sorting the search results presented to the user.

Other primitives are used for system internal needs. Here again, we invite the reader to consult the appendix for more details on the ontological primitives.

8.5 Ontology parts dedicated to documents

The memory of CoMMA is a documentary memory. Therefore, O'CoMMA includes a branch on Documents. We only give the top here, but this branch is deeper and can also be extended by users to include company specific documents.

document
abstract,
advertisement, publicity, promotion,
article,
book,
chart,
course, training document,
documentary file,
extracted document,
form,
illustration,
index,
index card,
ISSN holder document,
logo,
mail,
map,
memo,
minutes,
narration, story, account,
news,
newsgroup message, forum message,
official document,
presentation,
proceedings,
profile,
reference document,
report,
scenario analysis,
speech,
spreadsheet, spread sheet,
thesis,
transparency, slide,
transparency show,
trend analysis,
web page, web site,
web site,
see ontology for more details (definitions, sub concepts...)

Table 18 Extract from the documents branch

8.6 Ontology parts dedicated to the domain

There are several domains in O'CoMMA. We only presents the top of this branch that was extensively completed by end-users:

additional topic
agriculture,
aquaculture,
biology,
building,
cognitive sciences,
computer science,
earth observation,
economic science,
human science,
mathematics,
music,
physics,
social science,
telecommunications,
see ontology for more details (definitions, sub concepts...)

Table 19 Extract from the topic branch

Domain ontology, in the case of knowledge-based information systems, are also called topic ontologies. These branches are typically not reusable if the application domain changes. They usually require rework if the user changes, but the domain remains the same, in order to realign the conceptualisation of the ontology with the one of the new user. Existing ontologies may be reused to populate this part; the effectiveness of such approach is being studied by researchers, and an indicated field for such trial is the one of medicine. Again the use of natural language processing tools is interesting to analyse large corpora of the domain and populate the topic branch.

8.7 The hierarchy of properties

The current hierarchy of properties contains 79 relations and attributes with maximal depth of 2 subsumption links as given in Figure 64.

- some relation
  - has for number
    - phone number
    - fax number
    - mobile number
    - ISSN
    - number
    - ISRN
    - has for internal report number
    - ISBN
  - manage
  - administer
  - has for activity
  - created by
  - is interested by
    - has for work interest
    - has for personal interest
  - has for ontological entrance point
  - target
  - situated
  - include
  - colleague
  - designation
    - first name
    - name
    - title
  - related to
  - contain
  - concern
  - issued on the occasion of
  - service type
  - service provider
  - geographical area
  - network type
  - customer type
  - technology
  - refers monitoring to
  - assist
  - address
  - indication
  - birth date
  - employed by

employment contract
had for participant
organized by
beginning
end
has for perception mode
has for representation system
has for storage format
has for medium
has for origin
has for diffusion right
alternative document
summary
comments
creation date
keyword
sleeping partner
trainee originating from
extracted from
supervised by
domain
type
source
reliability
see also
hiring date
visited document
visitor
first visit
last visit
visit count
rating given
pushed document
description of a pattern
RDF string
has for favorite query
has for favorite annotation

Figure 64 Hierarchy of properties

A depth of a hierarchy of 2 is very low and hardly interesting. The relations are not structured enough and the taxonomy is too flat to bring visible improvements to the information retrieval inferences. That is why I started to divide for instance 'is interested in' into 'has for professional interest' and 'has for personal interest'. This work should be pursued since I believe taxonomy of relations can turn to be as important as the one of concepts for our scenarios. If the specialisation and generalisation inferences on properties are not as detailed as the one on notions, I am convinced of their usefulness e.g. *designation* enables to find any *resource* (a *book* with a *title*, a *person* with a *name*, a *project* with a *designation*, etc.) even if no information on the exact type is available. Finally, one has to take into account that n-ary relations are reified as concepts.

Specificity in the relation taxonomy is significantly lower than in the concept taxonomy. However its interest is not to be underestimated for the generalisation of requests.

8.8 End-Users' extensions

At the end of the first year of the CoMMA project, one of our partners, Telecom Italia, had to leave the project for "restructuring reasons". It was replaced by a new partner, CSTB (French building research centre playing the role of end-user for the same scenario). Although it was unfortunate for us to lose our Italian partner, this event enabled me to witness the appropriation of the ontology and its extension by the new partner. I reproduce here some of the tables created by CSTB to require new extensions and I discuss these extensions in comparison with the ontology structure depicted at the beginning of this chapter.

The first type of required extensions concerned the organisational modelling, to include external actor descriptions, and CSTB internal organisational concepts.

Notion	Definition
Partner	Any organisation with which the CSTB is associated and collaborates. The co-operation can be limited (e.g. for the realisation of a contract for a customer) or durable (mutual recognition of the evaluation procedures and test results, institutional partnership).
Competitor	*(No consensual definition given)*
Supervision authorities	CSTB is a public and commercial organisation under the supervision of the Housing Ministry.
Customer	Manufacturers, building contractor, engineering firms, architects and contracting authorities which form CSTB's customers.
Subsidiary company	*(No consensual definition given)*

Table 20 Organisational entities extensions

These extensions are mainly situated under the 'organisation' concept. Depending on the context, the same organisation can adopt those different status therefore, multi-instantiation will be used.

Notion	Definition
Management committee	Formed by the President, the Director, the Director of Research and Development Managing Board and the Technical Director.
Management board	Formed by all heads of services.
Department*	Thematic grouping of services (for example, the Security Department counts 2 services: "Structures" and "Fire").
Service*	Basic functional unit of CSTB. A service is part of a department and is composed by several divisions or poles.
Division*	Functional subdivision of a service. It counts generally 10 to 25 people.
Pole*	Functional subdivision of a service. It counts generally 2/3 to 7/8 people
Test Laboratory	It is attached to a service or a division.
** The definitions of these concepts are still unstable.*	

Table 21 Organisational parts extensions

These extensions are mainly situated under the 'organisation part' concept. An interesting problem shown here is that CSTB seems to have a problem to reach a consensual definition within its organisation.

As expected, the ontology part concerning the CSTB's expertise areas required a large domain extension. Complete new areas had to be described in the domain part of O'CoMMA. What is interesting here is that this domain ontology was provided by librarians of CSTB in the form of a nomenclature used for indexing documents in their library. An extract from this nomenclature in shown in Figure 65, and I apologise to non French readers, but this is the original collected document.

liste thématique de présentation des documents reçus à la documentation paris-champs
10 - sciences appliquées au bâtiment
11 essais - mesures - métrologie
12 acoustique
13 aérodynamique
14 résistance mécanique et stabilité
15 thermique, hygrothermique et éclairage
16 analyse et traitement de l'eau
17 mécanique des sols - génie civil
18 climatologie
19 énergie
20 - technique et technologie des ouvrages et matériaux
21 aménagements extérieurs - voirie, assainissement
22 gros œuvre
221 structure
222 enveloppe
223 toiture
224 façade
225 fondations
23 second œuvre
231 menuiserie (porte, fenêtre, volet)
232 isolation acoustique et thermique
233 revêtements sols et murs
234 cloisons
235 conduits et gaines
24 équipements
241 génie énergétique
242 équipements sanitaires
243 éclairage
244 Dominique - automatisme
25 matériaux de construction - produits de construction
26 pathologie - corrosion
27 chantier
30 - sciences économiques sociales et humaines
3100 prospective - recherche - innovation
3200 secteur du bâtiment
3210 urbanisme
3220 politique de la ville
3230 habitat
3240 politique et financement du logement
3300 sciences humaines
3400 économie
3410 économie de la construction
3420 économie de l'énergie
3430 données statistiques
3440 management
3500 acteurs de la construction
3510 métiers du bâtiment
3520 formation professionnelle
(...)

Figure 65 Extract from librarian nomenclature.

The librarian nomenclature is an extremely interesting document for the domain part ontology. It enabled to generate a first taxonomy very quickly. The only thing missing was the definitions attached to the themes; it appeared that somehow the lack of exact definition gives more flexibility to the indexing which is an aspect worth considering concerning the use and form of an ontology.

The following part is about the extension on persons and their characteristics.

Notion	Definition
Profession	There are 4 fields (defined in a 1954 decree), called "professions", structuring CSTB's activities: research, technical consulting, quality assessment and dissemination of knowledge. Those professions apply to each expertise areas (climatology, thermology….).
Research*	Work on innovative technologies and solutions to future needs and requirement of builders, manufacturers and end-users.
Consulting*	Scientific and technical consulting aims to find innovative solutions to the complex engineering problems that conventional methods are unable to solve.
Quality assessment	Evaluation, assessment and certification of building products and process.
Dissemination of knowledge	CSTB produces and disseminates information through various products and supports (internet, publications, training sessions…).
** These concepts are already included in the ontology, but in CSTB scenario they have a specific meaning.*	

Table 22 Profession extensions

Notion	Definition
External people	People from another company working for a limited period of time at CSTB.
Auditor-Inspector	To add to employee status.
Head of division	*(No consensual definition given)*
Head of pole	*(No consensual definition given)*
Laboratory manager	*(No consensual definition given)*
Post-doctorant	*(No consensual definition given)*
Archivists	To add in "Technical Monitoring Actor"

Table 23 Roles extensions

These extensions are mainly situated under the 'professional' and 'person' concepts. An interesting problem shown here is that some border notions of the ontology bottom and middle layers were judged incompatible with the home definitions of CSTB. This means adjustments had to be done and these concepts had to be customised. Some of these extensions concerned activities and therefore, went into the domain part of the ontology, most of the others went into the person branch of the ontology.

Additional extensions were finally required to describe the documents:

Notion	Definition
Final Report	Concludes and synthesizes the results of a research action or a consultancy contract.
Intermediate report	Punctual report produced at the end of each step of a research action or a consulting contract (state of the art, experiment…).
Activity report	Annual report written by each service in order to present its activities. It contains a list of the service's non confidential publications.
Research report	Report on a study founded by public authorities
Consulting report	Report on studies performed for clients. Most of them are confidential.
Training period report	Report written by a student at the end of its training period. Rarely confidential, these reports deal with very restricted area.
Standards	*(No consensual definition given)*
Patent	*(No consensual definition given)*
File	Thematic file, regularly updated and made with heterogeneous material (articles, references, synthesis…).
Training course book	Created by the « training » team of CSTB with the collaboration of other departments. These documents are distributed to participants of conferences, seminars, training session ... organized by CSTB.
Information forms	Written by the engineers and researchers to share their (informal) information.
Trends synthesis	Synthesis written by an expert on the trends of a technological area.
Confidential Document	The diffusion is restricted to a defined community. There are different levels of confidentiality for internal or external diffusion. A document can be composed of confidential parts or not confidential parts (for example: the reference and the abstract can be freely diffused, but not the integral text).
Public document	The diffusion is not subjected to any restriction.
Internal Document	Internal production (Information letter, Training course book, Reports, Intranet page …)
External Document	Books, articles…. acquired from an external source and written without the participation of any people from CSTB.
Paper Document	Paper support
Electronic Document	Electronic support
Extract	Document extracted from an other (ex.: communication of congress, article of journal…).

Table 24 Document extension

These extensions are purely extensions of the documentary part of the ontology. It shows a perfect case of reuse and extension of the ontology by the end-user.

A whole new part was also envisaged, but not modelled for lack of time. It was motivated by scenarios where the CSTB wanted 'information flows' to be captured so as to precisely identify which documents are preferentially manipulated by which actors at each step of the flow. Workflow and information-flow support were not part of the scope of the investigations carried out in CoMMA and therefore, this was left as a "nice to have". However, the reason why I mention it here is that it clearly appeared that in a complete solution to be deployed in real-world situations, explicit information flows descriptions and management would be a core functionality.

8.9 Discussion and abstraction

First of all, I would like to acknowledge the fact that, as the reader saw, O'CoMMA is a prototype ontology used for trials and to initialise the building of new ontologies rather than a fully mature and polished ontology. It is an experimental and learning tool.

The extension tables of the last section show a case where an end-user who did not participate to the original construction of O'CoMMA, can appropriate himself the ontology and customise it mainly by extension and revision of the middle layer.

The experience showed the usefulness of an existing ontology to initialise the construction of a customised ontology for a new application context. Through a guided visit of the ontology and a demonstration of the additional power in bringing to a solution, this enables designers to explain to the users why a co-operation is needed and what kind of contribution would be useful.

The use of the librarian nomenclature for domain extension proved an excellent move and the involvement of librarians in the ontology design process is a priceless asset. Librarians are aware of all the problems of thematic indexing of document. They have tools and models ready to be used and reused for ontology engineering. Likewise, in the technology monitoring process, observers and thematic area referent people were privileged interlocutors.

It is important to target existing profiles in the organisation roles that it is appropriate to involve in the ontology and memory creation and maintenance so as to ensure a moderate overcost (since the roles and infrastructures already exist) and a good expertise. They are to be actively implicated in the collection, definition and validation activities; they are both experts to be involved and pointers to other existing sources of relevant expertise.

In an other project with CSTB called APROBATIOM, a natural language processing tool (NOMINO) has been coupled together with classical data collection techniques. The feedback from the tools was not only interesting for us to collect candidate terms, but also for end-users who saw the emergence of a thematic axis they had not detected and not yet explicitly used for indexing. Moreover, the result of the analysis was grafted on O'CoMMA, to extend and customise it.

Natural language processing tools are extremely useful to customise an initial general ontology and populate application-specific parts such as the domain description.

Not all the extensions suggested by CSTB were accepted. For instance, extensions for security management on documents could be accepted, but even if the conceptual vocabulary was made available, it would have meant a tremendous software additional development to get the system to exploit these aspects. These computational extensions are feasible, but they were not part of the original scenarios and therefore, were completely ignored for trial.

By making explicit the conceptualisation, the ontology augments the adaptability of the solution. However its usefulness is also restricted by the computational consensus that was built above it and thus only extensions of the existing formal aspect already exploited by the system may be done without additional rework.

As the reader probably noticed, the quality of definitions is extremely heterogeneous. Sometimes only terms were proposed by end-users. In such a case I applied the following rules:
- the propositions must either be refused until the end-user proposes an associated definition, or be returned with a proposed definition and a request for confirmation.
- propositions that are not explicitly motivated by a realistic usage in the scenarios are refused.

Examples of discussions with the end-users are:
- "Auditor - Inspector": the question raised was that having a closer look at the definition proposed, it did not imply that the auditor be an employee, however the super class advocated by end-users was "employee"; this had to be discussed to remove the incoherence.
- "management committee": the concept was placed under "groups", but the definition stated that it was a group of individuals. Since the concept " a group of individuals" exists in the ontology, "management committee" was moved below.
- "division" and "department" were both labels of one concept; users expressed their will to separate them and new concepts were created.
- a list of "head of ..." was given by the users describing the different types of chiefs. They were restructured under a new concept "head / chief" created to represent what they have in common.
- a "Pole contains 2 to 8 persons" and "a division 10 to 25" ... the question raised was what is a group of 9 persons ?

Definitions were discussed and revised with users to eliminate:
- *incoherence*: a definition contradicts itself or another definition. Reaction: contradiction is sent back to the user and the use of the contradicted notions is suspended until an acceptable correction is provided. It is usually worth the differentia and the inheritance of each one of the notions to detect the source of the incoherence.
- *duplication*: a definition provides no distinction with another existing definition. Reaction: the user is asked to provide the differentia or accept the merging (usually simply by adding terms to the synonyms of the existing definition).
- *circular definitions*: a definition references itself *i.e.,* the notion being defined appears in its own definition. Reaction: the definition is rejected and a reformulation is required suggesting to use direct subsuming notions as primitives.
- *imprecision*: the concept is placed at a high level in the hierarchy while it uses some more precise primitives. Reaction: the user is asked to choose between a more precise position or a less restrictive definition. It may also happen that the imprecision is due to the lack of more precise primitives in the ontology, then the addition of such primitives is suggested to the user together with the new positioning of the notion.
- *non exhaustivity*: the definitions of the concepts denote disjoint classes and there exist some cases that are not covered or the proposed definitions simply do not provide the primitives for some scenario cases. Reaction: the user is prompted the cases and asked to clarify if the missing cases can occur; if they can occur, additional primitives are required, and if they cannot occur, explanations are added to scenarios and definitions to save the design rationale.
- *fuzziness*: the context of the ontology does not enable us to ensure that the interpretation of the definition will be unique. Reaction: cases of ambiguity have to be identified and the choice of introducing several notions to cover the different cases or to restrict the definition to one special context must be made.

Finally, an example of incompatibility between two ontologies was found: one of the end-user organisation defined "an *index card* as a *news*" and another defined "an *index card* as a *report*", *report* and *news* being mutually exclusive for each of these users. Asking them to discuss the problem together, "they only agreed to conclude that they disagree"... This means the ontologies of these end-users are compatible to a certain point (the limit marking the domain on which they could interact) and that two ontologies or two views are needed below that point.

This concludes the part describing the corporate semantic web and ontology approach. Now that the information landscape is semantically annotated, the issue of the memory concerning the persistent storage and indexing of resources has been addressed. The second issue is the one of a software architecture capturing and diffusing the knowledge stored and indexed in the memory. In CoMMA, it is the role of the multi-agent system. The architecture and the design of this system are described in the following part.

Part - III

Multi-agent system for memory management

The Law of Requisite Variety:
"Only variety can destroy variety"
— William Ross Ashby

9 Design rationale of the multi-agent architecture

The law of requisite variety say that a regulator must have as much or more variety, than the system it regulates. This, in a way, is an incentive to consider distributed artificial intelligent systems to tackle problems of distributed knowledge.

This chapter presents the design rationale that we followed in CoMMA to obtain a multi-agent architecture supporting both corporate memory management scenarios we envisaged. We were especially interested in the annotation storage and retrieval through push and pull techniques.

First, I shall explain every main stage of the organisational top-down analysis of functionalities. We shall see that through the identification of societies dedicated to some resources of social task and the analysis of their organisation, we get down to the point where roles and interactions can be identified. Then, I shall describe the characteristics and documentation of roles and protocols supporting each sub-society. I shall explain how these roles were implemented into behaviours attached to agent types and finally, I shall briefly discuss about the deployment of agent instances to form a configuration depending on the organisational internal network.

9.1 From macroscopic to microscopic

The functional requirements of the system, as identified in the scenario analysis, do not simply map to some agent functionalities, but they influence and are finally diluted in the dynamic social interactions of agents and the set of abilities, roles and behaviours attached to them. This section explains how we went from the system requirements expressed at the societal level from the use-cases, down to the point where agent roles could be identified.

9.1.1 Architecture versus configuration

The MAIS architecture is a structure that portrays the different kinds of agencies existing in an agent society and the relationships among them. A configuration is an instantiation of an architecture with a chosen arrangement and an appropriate number of agents of each type. One given architecture can lead to several configurations.

In the case of a multi-agent corporate memory system, the configuration depends on the topography and context of the place where the system is rolled out (in particular: organisational layout, network topography, stakeholders location), thus it must adapt to this information landscape and change with it. The architecture must be designed so that the set of possible configurations covers the different corporate organisational layouts foreseeable. The configuration description will be discussed at the end of this chapter.

The architectural description is studied and fixed when designing the MAS. The architectural analysis starts from the highest level of abstraction of the system (*i.e.,* the society) and by successive refinements (*i.e.,* nested sub-societies), it goes down to the point where the needed agent roles and interactions can be identified.

We followed a top-down analysis based on a decomposition and allocation of the organisational functions.

Considering the system functionalities, we identified four dedicated sub-societies of agents:
- The sub-society dedicated to Ontology and Organisational model
- Annotation-dedicated sub-society
- User-dedicated sub-society
- Connection-dedicated sub-society managing yellow and white pages.

As shown in Figure 66, a first high-level acquaintance graph can be produced, whose nodes are agent sub-societies and whose links denote the initial inter-society acquaintances existing. It is clear that the interconnection sub-society is the backbone enabling agents to get acquainted.

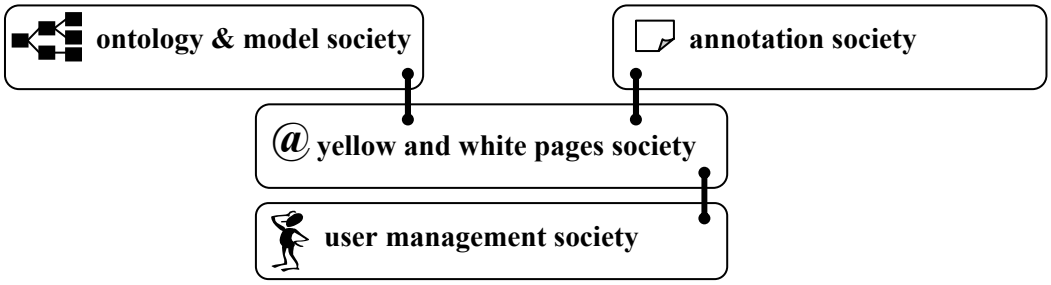


Figure 66 Sub-societies acquaintance graph

At this level we are completely independent of a chosen architecture, yet we are already influenced by the panel of possible divisions of functionalities that may be imposed by the different foreseeable configurations.

Refining Figure 66, nodes were mapped to agent roles, and then in a real configuration, roles are played by agent instances and edges become acquaintance relationships along which agent interactions take place. This last level is completely tied to a configuration and described in a deployment diagram. The interaction protocols an agent gets involved in, derive from its social roles and the acquaintance connections among the agents derive from both the MAS architecture and the use case scenarios.

The following section is an intermediary step in the decomposition and identifies the conceivable structures for the various sub-societies.

9.1.2 Organising sub-societies

As noted in the state of the art, there exist organisational structure patterns that can be abstracted and reused. The internal organisation of each sub-society is mainly the result of the range of configuration envisaged and the associated issues: being able to instantiate the structure of the various sub-societies for foreseeable configuration is the key to achieve flexibility in the deployment phase. Analysing the sub-societies dedicated to the management of some information resources ("Ontology and model", "Annotation" and "Yellow and white pages") we found that there was a recurrent set of possible internal organisations for these sub-societies.

9.1.2.1 Hierarchical society: a separated roles structure

As illustrated in Figure 67, the hierarchical organisation distinguishes between two kinds of roles:
- The *representative role*: the agent that plays that role is a mediator between its society and the rest of the MAS. It deals with the external requests. If needed, it breaks them up into several sub-requests. It contacts resource-dedicated agents and delegates to them. Finally, it compiles the possibly partial results to answer the external requester.
- The *resource-dedicated role*: the agent that plays that role is dedicated to a local resource repository and contributes to solve the requests it receives from the representative as much as it can with its local resources.

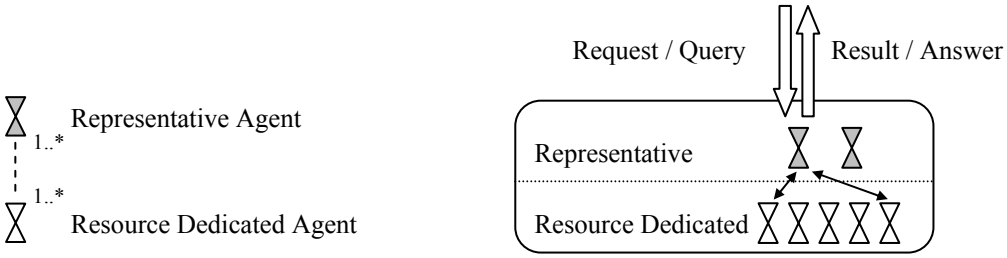


Figure 67 Hierarchical society organisation

In this sub-society, the resources are distributed over the resource-dedicated agents; more precisely, at least one resource agent is present on a node where a given resource must be accessed. This enables the geographical localisation of the information repositories and their maintenance which can be important if the agent must locally wrap a source that was not designed to be shared before. For example, financial annotations could be located in the finance department and therefore, ease the maintenance by a local expert and maybe avoid network jamming if we consider that an important part of the queries concerning finance will be issued by people actually working in the finance department. These annotations may be produced by a legacy system that needs to be wrapped in an agent or coupled to a transducer agent.

The mediator can either be:
- a *generic manager*: just co-ordinating with no skills and no role concerning the processing of the information resources
- a *specialised manager*: having skills in the domain of the information resources managed by this society, and being in charge for instance of intelligently breaking down the requests and compiling the results, thus enabling a specialised handling of the problems raised by the distribution of the resources.

In this society, the workload can easily be distributed if we consider that the resource agents manage only the resource they locally have, leaving the decomposition, the merging, and in general the society management tasks to the resource mediator agents. Thus resource mediators can be placed on powerful servers that do not necessarily hold an information resources repository and balance the workload over other nodes. This is due to the fact that we distributed skills over two roles. Specialised roles allow the work distribution, but on the other hand this approach is much more network-consuming than the others since role distribution requires additional co-operation interactions.

9.1.2.2 Peer-to-peer society: an egalitarian roles structure

As shown in Figure 68, a peer-to-peer organisation sets up egalitarian relationships between agents instantiating a unique kind of role. Roles are not distributed, but completely redundant: any agent can be contacted from outside the society to handle a request concerning the resource type its society is dedicated to. It will then have to co-operate with its peers in order to efficiently fulfil the request.

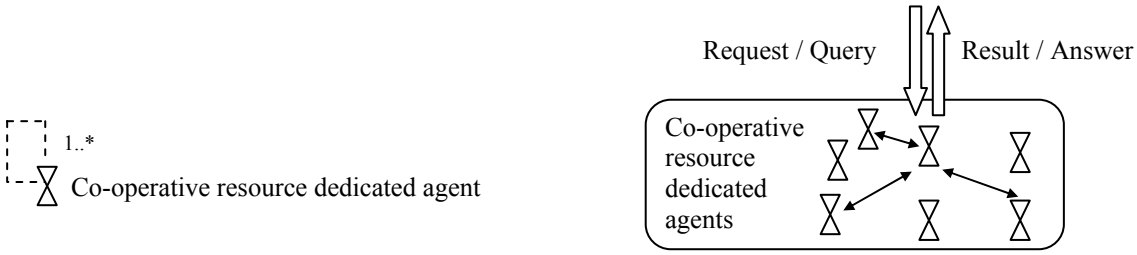


Figure 68 Peer-to-peer society organisation

Agents are specialised only through the content of the local resource repository they are attached to. The role and attached skills are identical for each agent playing this role. There is only one role merging the two previous roles (representative and resource dedicated). Coalitions based on a co-operation protocol are formed to solve external queries. The workload is less distributed than in the previous case, but the network-load may be decreased.

9.1.2.3 Clone society: a full replication structure

As shown in Figure 69, a replication organisation is a subtype of the previous case: neither the roles nor the repository content are distributed. Each agent keeps up to date a complete copy of all the resources and is able to solve any request by itself.

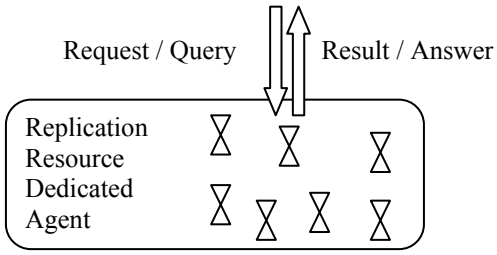


Figure 69 Replication society organisation

Therefore, the only social interactions that exist are for content updates propagation. The workload is even less distributed than in the previous case and the contents has to be replicated everywhere an agent runs, which can be an unacceptable constraint for some types of information resources repositories (e.g. annotations archives). On the other hand, there is no specialisation, therefore, the system is highly redundant and thus more robust, and the network use is minimal when dealing with a request. The only role left from the previous cases is the resource-dedicated one.

9.1.2.4 Choosing an organisation

Depending on the type of tasks to be performed, the size and complexity of the resources manipulated, a sub-society organisation will be preferred to another.

Each one these organisations can be implemented using different and multiple protocols to set-up the social interaction that positions the roles in the acquaintance network and enable the effective co-operation of the agents.

In the following section, I detail and motivate our choices for each one of the sub-societies identified in the multi-agent architecture of CoMMA

9.1.3 Sub-society dedicated to ontology and organisational model

"Ontology agents are essential for interoperation. They provide a common context as a semantic grounding, which agents can then use to relate their individual terminologies; provide (remote) access to multiple ontologies; and manage the distributed evolution and growth of ontologies." [Singh and Huhns, 1999].

Agents dedicated to the ontology and the corporate model are provider agents. They are concerned with the ontology and model exploitation aspects of the information retrieval activities. They provide downloads, updates and querying mechanisms for other agents, on the hierarchy of concepts and the description of the organisation.

For this sub-society, the three types of organisations are conceivable:
- In a *hierarchical organisation*, there are a Mediator role (in charge of resolving external requests) and an Archivist role (in charge of a part or a view of the ontology or model).
- In a *peer-to-peer organisation*, there is a single role: a co-operative archivist and solver role.
- In a *replication organisation*, the complete ontology and model of the system are replicated so that each agent has a complete copy of the information and can resolve requests by itself. There is only one role: a stand-alone archivist role.

The last choice is acceptable when the ontology is stable and when a consensus is reached by the users so that the ontological commitment is centralised and the global ontology is updated and propagated over the agent society. This option is implemented in the current prototype of CoMMA.

The remaining options are interesting if the ontology/model is large or changes quite often and if the system must support the ontological consensus process; in that case, this society can support the break-up of the ontology/model and maintain the coherence between the different repositories as in the FRODO project [Elst and Abecker, 2001].

The actual choice of CoMMA is of course only acceptable in the context of a prototype system focusing on the annotation management scenarios. As I shall discuss in the perspectives, the other options, and especially the egalitarian one, are much more likely to be chosen in the case of a solution implementing ontological consensus management functionality.

9.1.4 Annotation dedicated sub-society

Document-dedicated agents are typical provider agents. They are not wrapper agents working on unstructured heterogeneous sources of data; they are concerned with the exploitation of the annotations structuring the corporate memory, they search and retrieve the references matching a given query.

Concerning this sub-society, only the two first types are conceivable:
- In a *hierarchical organisation*, there are a Annotation Mediator role handling external requests and an Annotation Archivist role wrapping annotation bases.
- In a *peer-to-peer organisation*, there is a co-operative Annotation Archivist role combining both previous roles.
- A *replication organisation* is not realistic because it would imply to replicate a full image of the annotations of the memory over each resource agent. This condition is obviously not acceptable since we want to wrap the source of annotations distributed over an intranet where they are generated.

The current CoMMA system opted for the first type of society. It is extremely interesting to de-couple the archivist role and the mediator role because it enables us to make the system extensible and to easily introduce different types of archivists (annotation archivists, Database wrappers, Web page miners, etc.) without changing anything to the architecture.

The Annotation Mediator typically provides other societies with its services to solve their queries and it requests the services of the resource agents to effectively solve them:
1. It breaks the requests and contacts the relevant Annotation Archivists at each stage of the resolution process to get partial results.
2. It compiles the partial results and builds a final result.

The Annotation Archivists are attached to a local annotation repository. When they receive a request, they try to obtain at least partial results from their repository to enable the mediator to handle results distributed over several information sources.

9.1.5 Interconnection dedicated sub-society

Agents from this sub-society are in charge of matchmaking other agents using their respective needs and service provider descriptions. As noted in [Hayden *et al.*, 1999] "the adoption of an appropriate co-ordination mechanism is pivotal in the design of multi-agent system architectures".

The use of middle agents such as brokers, facilitators, mediators or matchmakers appears to be the most frequently implemented co-ordination mechanism. Each provider must first register itself with at least one middle agent and advertise its capabilities. Requests are then matched to these descriptions to find which agent can provide a required service. Middle agents allow to de-couple the service providers and the service requesters; each agent is no longer obliged to maintain a complete acquaintance list of all other providers it may need to contact. Instead, it only has to know an agent providing Yellow Pages services and may be an agent providing White Pages services for locating appropriate agents with appropriate capabilities.

The CoMMA system is implemented using the JADE platform [Bellifemine *et al.*, 2001] that provides an agent type called Directory Facilitator and an other agent type called Agent Management System:

- The *Directory Facilitators* (DFs) are federable to build a peer-to-peer society and are in charge of managing Yellow Pages. Yellow Pages service consists of providing the address of an agent giving one or more of its capabilities. According to FIPA specifications, DFs offer agent identifiers matching the service description and the ontology specified in a pattern. Thus DF are matchmakers identifying relevant providers and returning the selection of candidates to the requester. The result of the matchmaking can be further refined in a second stage. For instance, as detailed in chapter 10, the Annotation Mediator requires statistics from the Annotation Archivists to know what kind of annotations their archives contain so as to decide when to appeal to them during the distributed solving process.
- The White Pages are managed by the *Agent Management System* (AMS) agent. White Pages service consists of providing the address of an agent giving its name.

9.1.6 User dedicated sub-society

The agents from this sub-society are concerned with the interface, the monitoring, the assistance and the adaptation to users. They are, typically, requester agents. Because they are not related to one resource like others (ontology, annotation or Yellow Pages), they cannot be studied using the typology I proposed at the beginning. Roles defined in this sub-society and their distribution depend on additional functional specifications of the system.

I distinguish two recurrent roles in this type of sub-society:

- The first role is the user *interface management* itself: dialogue with the users to enable them to express their request, to refine them and to present results in a comprehensive format. If this role is implemented as an independent agent, this agent may or may not be online (no temporal continuity) depending on whether or not the user is logged onto the MAS system.
- The second recurrent role concerns the *management of user's profile*. If implemented as an independent agent, user profile agents are just like archivist agents except their annotations are about users. They are ever running agents (temporal continuity), and enable the profiles to be used not only for interface purposes, but also for learning techniques, pro-active searches, etc.

These two recurrent roles in MAS systems may be merged into one agent or more roles may be added to implement specific functionalities of the system.

I shall start from the two functionalities expected for the management of users:

- management of an interface enabling the interaction between the user and the system; this role is called *Interface Controller*.
- updating (*i.e.,* customising and learning) and exploiting the user's profile when the user is logged on to the system; this role is called *User Profile Manager*.

9.1.6.1 Possible options

For the architecture, we have two options: there can be one User Profile Manager per Interface Controller, that is a pair of these two roles should be played per user; or a single User Profile Manager can manage several profiles at the same time. In the first case, a User Profile Manager would be instantiated when the Interface Controller is created that is to say at login time. In the second case, User Profile Manager would be persistent and at login time, the freshly created Interface Controller would negotiate with running User Profile Managers which one of them is going to manage the profile of its user during the session. For the interactions between these two roles and with the rest of the systems, two basic approaches can be identified:

- First option: *Reporting*. The Interface Controller reports every event to the User Profile Manager, but contacts by itself the other agents it needs to collaborate with.
- Second option: *Intermediate*. The Interface Controller sends everything to the User Profile Manager; the User Profile Manager analyses the messages and forwards them when needed.

Both above approaches raise problems:

- The first option does not allow the User Profile Manager to modify the Interface Controller messages and this is a problem if the User Profile Manager wants to enrich user queries or results exploiting the contextual or profile information it gathered from the user.
- In the second option, the User Profile Manager may have to forward messages it is not interested in, for instance, when the Interface Controller is requesting the download of an ontology.

So we naturally envisaged the hybrid situation where the Interface Controller decides whether the message is to be sent directly to the specialised agent or if it must go through the User Profile Manager first. This decision is based on the nature and content of the message.

9.1.6.2 User profile storage

Then, comes the problem of storing the user profile. Since the Interface Controller is not continuously running, user profile storage cannot be one of its roles. If the User Profile Manager is also unique and created at login time (first option we envisaged) then storage cannot be one of its roles either. So, the User Profile Manager would have to be continuously running in a fixed location, in order to be always available. On the other hand, if we consider that a CoMMA user could log into the system from many different locations, the intense traffic between Interface Controller and a distant User Profile Manager could burden the network a lot. In summary, user profile storage responsibility requires a fixed and persistent agent to play the role, whereas reducing network load requires an agent that lives in the proximity of the Interface Controller to play the role of User Profile Manager at least for login time.

To solve this conflict, a third solution was imagined: a *User Profile Archivist* agent could be in charge of the profile storage and access. At login, the Interface Controller contacts the nearest User Profile Manager (network load would be saved since the traffic between these two agents is likely to be significant) and the User Profile Manager retrieves information about the user from the Profile Archivist responsible for that user. When the user logs off, the profile is saved.

From a configuration point of view, at a certain location, a single User Profile Archivist will manage the profile of all the users having that location as their home location. On the other hand, at a certain location, a single User Profile Manager will manage all the users having that location as their current location.

One could also imagine the possibility of a profile-dedicated sub-society since user profiles can be viewed as annotations about people. However, there is one and only one profile per user, and this profile is not distributed over the intranet, therefore, we do not need a complete sub-society. If the profiles were to be distributed, many more constraints would have to be taken into account (e.g. distributed updates). However to take into account the profiles when solving queries (e.g. find documents written by people interested in Knowledge Management), the profile base may also be exploited by an Annotation Archivist as additional annotations of the memory.

The CoMMA system also takes the initiative in a proactive way, exploiting its knowledge of the user to anticipate her or his needs (this is especially true in the Technology Monitoring scenario). The profile is stored by the User Profile Archivist, so either the User Profile Archivist itself is in charge of the proactive analysis of the user profile or the User Profile Archivist contracts a *User Profile Processor* agent to handle it.

The decision about whether to introduce User Profile Processor or just use existing User Profile Archivist is affected from workload balancing issues. Relying on the User Profile Archivist for proactive behaviour would limit the number of agents in the system, while introducing special purpose User Profile Processor agents would distribute more the work, increasing the separation of concerns as a side effect. The User Profile Processor agent, if used, would need to communicate with the User Profile Archivist agent (to retrieve the user profile), and probably also with the agents in the document dedicated sub-society, in order to be able to perform proactive queries on the corporate memory.

9.1.6.3 Other possible roles for interest group management

We already envisaged up to four roles. The profusion of possible options gives an idea of the debates that took place between the designers and the end-users. However I also give here additional agent roles that were never considered for the CoMMA prototype, but were discussed as potentially interesting for extensions of the project. They participate in the proactive user support.

If we want users to register themselves in interest groups working on the same principle as news group, then we could introduce a *Public Interest Group Monitor* role analysing annotation submission events notified by Annotation Mediators; these events are filtered based upon the definition of the interest group. If a new document seems relevant for an interest group, it could be pushed to the group members. Another strategy for proactive information push would be to represent an interest group with a set of recurrent queries included in the definition of the interest; the Public Interest Group Monitor would then periodically issue these queries and push the changes in the results to the group members or keep up-to-date a document summarising these results.

Another approach is to cluster the users, without them registering anywhere as it is done in some collaborative filtering systems. Then, we would need an *Emergent Interest Detector* role looking for implicit communities of interests analysing the users' profile. When an Emergent Interest Detector determines that a user belongs to an implicit community of interest, it can use the profile of the other members to make suggestions. Since this clustering process is performed without any explicit permission from the user, privacy issues will have to be considered (e.g. the user can disable the feature, or it can give private status to a part of his/her preferences so that the Emergent Interest Detector does not use them).

9.1.7 Overview of the sub-societies

For the sake of the feasibility, not all the roles of the user-dedicated societies have been considered. The Figure 70 shows the sub-societies with the new roles identified.

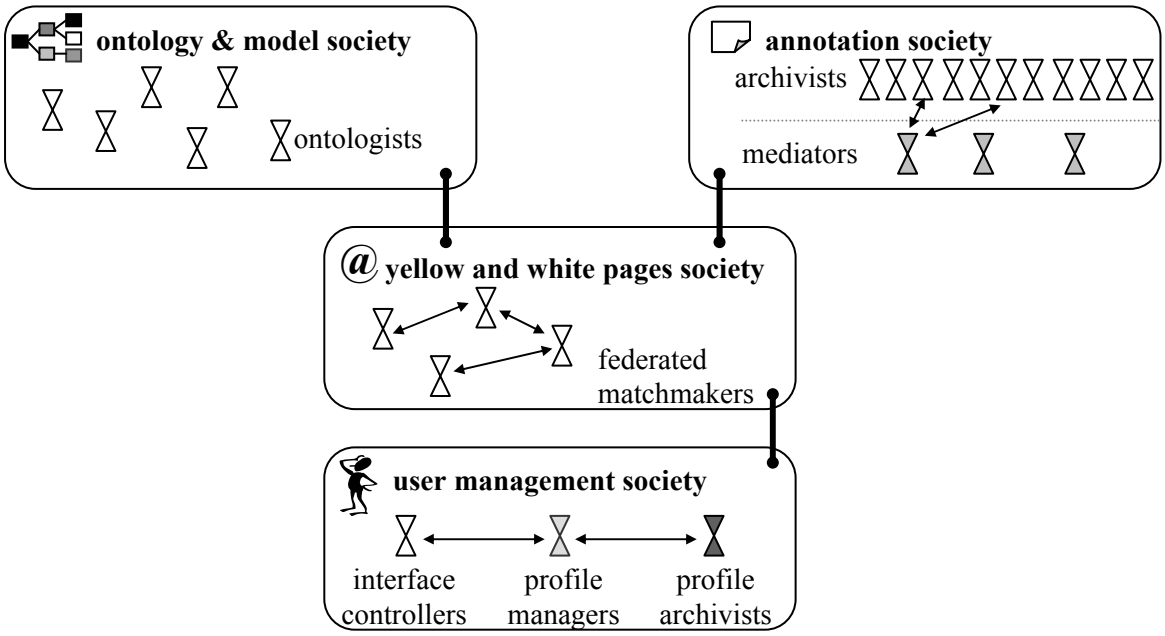


Figure 70 Sub-societies and their internal organisation

Eight roles have been accepted for the prototype and trial: Ontology Archivist, Corporate Model Archivist, Annotation Archivist, Annotation Mediator, Directory Facilitator, Interface Controller, User Profile Manager, User Profile Archivist.

The User Profile Processor has been merged with other roles as detailed later. These roles are the turning point in the design process: until now we have been considering roles from a social perspective, but the following section will consider them from the individual perspective of the agents playing these roles.

9.2 Roles, Interactions and behaviours

From the architecture analysis, I now derive the characteristics of the identified roles, their interactions and I discuss the implementation of the corresponding behaviours in a set of agent types.

9.2.1 Accepted roles

Roles represent the position of an agent in a society and the responsibilities and activities assigned to this position and expected by others to be fulfilled. In the design junction between the micro-level of agents and the macro-level of the MAS, the role analysis is a turning-point. The previous part identified the roles inside the societies, we now shift to the micro-level considering the system from the point of view of an agent playing each one of the roles.

As advocated by my colleagues of the University of Parma (members of the CoMMA project), roles have been described using the role cards of E. Kendall [Kendall, 1999]. The definitions of the facets of an agent role are:

role model	name and context
responsibilities	services, tasks, goals, obligations, interdictions
collaborators	roles it interacts with
external interfaces	access to external services or resources
relationships	aggregation, specialisation, generalisation, role sequences
expertise	ontology, inferencing, problem solving knowledge
interactions	protocol, conflict resolution, knowledge of why other roles are related, permissions
others	-

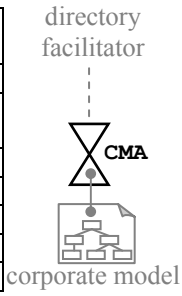
9.2.1.1 Ontology Archivist

role model	Ontology Archivist role in the Ontology and Model society
responsibilities	store and retrieve O'CoMMA
collaborators	Directory facilitator, Interface Controller, User Profile Manager, User Profile Archivist, Annotation Mediator, Annotation Archivist, Corporate Model Archivist
external interfaces	RDFS schemas manipulation interface
relationships	-
expertise	ontology management an querying
interactions	Query-Ref, Request and Subscribe; FIPA ACL
others	-

The role of Ontology Archivist (OA) has the duty to store and retrieve information to/from the O'CoMMA repository, saved as an RDF schema file. It is part of the society dedicated to the ontology and corporate model. Only this role has access to the ontology repository. The Ontology Archivist interacts with all the other roles because all of them need the ontology for their tasks, except the Directory Facilitator with which the Ontology Archivist interacts to register and deregister its services. The interaction with these agents is done according to FIPA request protocols for downloads and updates on the ontology and FIPA subscribe to be notified of any change in the ontology. It also initiates FIPA-Request protocol to register and deregister itself with a Directory Facilitator.

9.2.1.2 Corporate Model Archivist

role model	Corporate Model Archivist role in the Ontology and Model society
responsibilities	store, query, retrieve the organisational model
collaborators	Directory facilitator, Interface Controller, Annotation Mediator, Ontology Archivist
external interfaces	RDF annotation manipulation interface
relationships	composition: management and annotation archivist
expertise	annotation management an querying
interactions	Query-Ref, Request, Subscribe; FIPA ACL
others	-



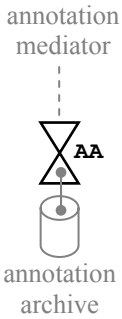
The Corporate Model Archivist (CMA) has the duty to manage and solve queries about the corporate model composed of RDF annotations about the organisational entities. It is part of the society dedicated to the ontology and corporate model. Only this agent has access to the corporate model repository, however this role is a composition of two roles:

- provide *management* (update rights, etc.) and download service (for roles needing to download the whole corporate model such as the Interface Controller).
- provide *query solving* engine to make the corporate model participate to the distributed query solving process over the whole memory; this role corresponds to the Annotation Archivist with the restriction of not accepting new annotations since this responsibility requires the corporate model management skills of the first part of the role.

The Corporate Model Archivist role interacts with the Directory Facilitator to register and deregister itself its services and search for collaborators using FIPA Request protocol. The first part of its role interacts with the Interface Controller for downloads and updates using FIPA Query-ref and FIPA Request. The second part of the roles interacts with the Annotation Mediator for solving queries using FIPA Query-ref and with User Profile Manager to notify changes in the model using FIPA Subscribe. It interacts with the Ontology Archivist to get the ontology using FIPA Request.

9.2.1.3 Annotation Archivist

role model	Annotation Archivist role in the Annotation-dedicated society
responsibilities	store and query the annotations of the memory
collaborators	Directory facilitator, Annotation Mediator, Ontology Archivist
external interfaces	RDF annotation manipulation interface
relationships	also part of the roles in Corporate Model Archivist and User Profile Archivist
expertise	annotation archiving an querying
interactions	Query-Ref, Contract-Net; FIPA ACL
others	-

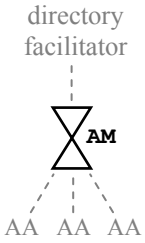


The Annotation Archivist (AA) has the duty to store annotations of the corporate memory and solve queries using its local archive. It is part of the annotation-dedicated society. Only this agent has access to the repositories of annotations about the document of the memory. The Annotation Archivist interacts with the Annotation Mediator to register and deregister itself and search for collaborators using FIPA-Request protocol, to solve queries using Query-Ref protocol, to archive annotations using Contract-Net protocol [Davis and Smith, 1983]. It interacts with the Ontology Archivist using FIPA Request to obtain the ontology. It interacts with the Directory Facilitator to find the Annotation Mediator and the Ontology Archivist.

(More details about this role are given in chapter 10).

9.2.1.4 Annotation Mediator

role model	Annotation Mediator role in the Annotation-dedicated society
responsibilities	handle distribution of annotations over the archivists both for new annotation submissions and query solving processes
collaborators	Directory facilitator, User Profile Manager, Ontology Archivist, Annotation Archivist, Corporate Model Archivist
external interfaces	RDF annotation manipulation interface
relationships	-
expertise	query and submission management
interactions	Query-Ref, Contract-Net, Subscribe, Request; FIPA ACL
others	-

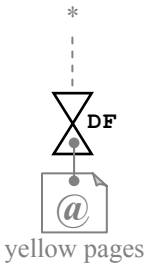


The Annotation Mediator (AM) has the duty to manage distributed query solving and annotation archiving requests, by mediating among the Annotation Archivists. The Annotation Mediator interacts with the Directory Facilitator to register and deregister its services and search for collaborators using the FIPA Request protocol. It interacts with the Annotation Archivist to solve the submitted queries using FIPA Query-Ref protocol and to allocate newly submitted annotations using Contract-net protocol. It interacts with the User Profile Manager that submits the queries on behalf of the users, using a FIPA Query-Ref protocol, submits new annotations using the FIPA Request protocol and registers for new annotation notifications using FIPA Subscribe. Finally, it interacts with Ontology Archivist to get the O'CoMMA ontology using FIPA Request.

(More details about this role are given in chapter 10).

9.2.1.5 Directory Facilitator

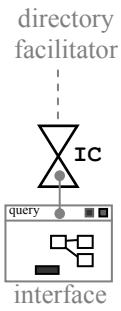
role model	Directory Facilitator role in the Yellow and White pages society
responsibilities	handles matchmaking among agents providing registering and de-registering services as well as service querying
collaborators	Ontology Archivist, Interface Controller, User Profile Manager, User Profile Archivist, Annotation Mediator, Annotation Archivist, Corporate Model Archivist
external interfaces	-
relationships	-
expertise	FIPA Service description management
interactions	Request; FIPA ACL, FIPA-Agent-Management ontology
others	(Provided by JADE)



The Directory Facilitator is the agent in charge of maintaining system yellow pages where agents may register themselves and their capabilities and to which they can submit search requests about service descriptions to find collaborators. This role is provided and implemented in any FIPA-compliant platform [FIPA]. Therefore, in JADE [Bellifemine *et al.*, 2001], any agent of the system using the Directory Facilitator services can register itself and access to the address of all the registered agents on the basis of queries about agent capabilities. It uses the FIPA-Agent-Management ontology to respond to FIPA-Request protocol with FIPA-Agent-Management content to perform: register, deregister, modify and search actions. It also initiates FIPA-Request protocol with FIPA-Agent-Management content to federate itself with another Directory Facilitator.

9.2.1.6 Interface Controller

role model	Interface Controller role in the user-dedicated society
responsibilities	handles interface between the user and the rest of the multi-agent system
collaborators	Ontology Archivist, User Profile Manager, Corporate Model Archivist, Directory Facilitator
external interfaces	graphic interfaces, HTML browser and XLM/XSLT engines
relationships	-
expertise	User interactions management
interactions	Request; FIPA ACL
others	not persistently running



The Interface Controller role is in charge of the system front end, working in contact with the user. It handles the GUI to make the user look like just another agent to the rest of the system: as soon as the GUI layer is crossed, all the information passing happens by means of FIPA ACL messages. This agent in the system that is not running persistently, with its lifetime limited to a single session: the Interface Controller starts when the user logs into the CoMMA system and shuts itself down when the user logs out. When the Interface Controller starts up, it uses the yellow pages services provided by the Directory Facilitator to register itself and get acquainted with all the necessary agents using FIPA Request protocol. During system usage, the Interface Controller interacts directly with the agents belonging to the ontology sub-society using FIPA Request. For other interactions, it goes through the User Profile Manager agent to deal with agents of the user sub-society and of the document sub-society: it initiates FIPA-Request protocol to login to the system and to require the User Profile Manager to get the user's profile from the User Profile Archivist; and it initiates FIPA-Request protocol with its User Profile Manager to submit user's queries or new annotations that should be forwarded to the system.

9.2.1.7 User Profile Manager

role model	User Profile Manager role in the user-dedicated society
responsibilities	handles customisation and adaptation in the user profile
collaborators	Ontology Archivist, Interface Controller, Annotation Mediator, User Profile Archivist, Directory Facilitator
external interfaces	machine learning techniques libraries
relationships	(partly absorbed the User profile processor)
expertise	learning from usage patterns
interactions	Request; FIPA ACL
others	-

A User Profile Manager agent is responsible for all the users currently connected to the system from within its jurisdiction domain; the Directory Facilitator gives the address of the User Profile Manager to any Interface Controller requiring it. This means that the whole corporate network is partitioned in subsets and there is one and only one User Profile Manager for each subset. The User Profile Manager role has the responsibility to exploit and improve the information contained in the user profile to enhance system performance e.g. by enriching user profile with a newly detected interest.

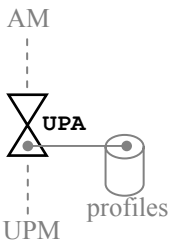
The User Profile Manager interacts with the Interface Controller through FIPA Request protocol. The User Profile Manager interacts with the Directory Facilitator to register and deregister its services and search for collaborators using the FIPA Request protocol. It interacts with the Annotation Mediator to solve queries using the FIPA Query-ref protocol, and to archive a newly submitted annotation using the FIPA Request protocol. It interacts with the User Profile Archivist to retrieve the profile of a user logging and to save the profile of user logging off using the FIPA Request protocol.

The User Profile Processor which has the duty to perform proactive queries on the corporate memory on the basis of user profiles was replaced by a subscription of the User Profile Manager to the Annotation Mediator. Thus the User Profile Manager gets the new annotations and push them to the User Profile Archivist to store suggestions in the profiles or optionally push them to the Interface Controller a user is logged to.

Emergent Interest Detector and Public Interest Group Monitor were not implemented because it turned out that the push technology and the user profiles enable a *de facto* emergence of communities sufficient for the first prototype of CoMMA. More complex mechanisms could be added later.

9.2.1.8 User Profile Archivist

role model	User Profile Archivist role in the user-dedicated society
responsibilities	handles storage and access in the user profile
collaborators	Ontology Archivist, Annotation Mediator, User Profile Manager, Directory Facilitator
external interfaces	RDF annotation manipulation interface
relationships	composition: management and annotation archivist
expertise	storage, access right and retrieval of profiles
interactions	Request; FIPA ACL
others	-



The User Profile Archivist (UPA) has the duty to manage and exploit user profiles from a CoMMA user profile repository maintaining the profile of the users belonging to a specific sub-net of the enterprise intranet. Only this agent has the access to the user profile repository of a specific sub-net.

As in the case of the Corporate Model Archivist, two sub-roles were identified:
- provide *management* (update rights, etc.) and secure the archiving and retrieval functionality required at login and logout time (by the User Profile Manager at the Request of a new Interface Controller).
- provide *query solving* engine to make the user profiles participate to the distributed query solving process over the whole memory; this role corresponds to the Annotation Archivist with the restriction of not accepting new annotations since this responsibility is the one of the management part of the role.

How the profiles for all the users are divided among the User Profile Archivist agents depends on the actual structure of the corporation where the CoMMA MAS is deployed.

The User Profile Archivist interacts with the User Profile Manager responding to FIPA-Request protocol for profile retrieval, update and new annotation matching and push. It initiates FIPA-Request protocol to register and deregister itself with a Directory Facilitator.

Remark: the roles presented here were extensively discussed within the CoMMA consortium; not all the details and options were reported here and a lot of questions could be further discussed.

9.2.2 Characterising and comparing roles

In Table 5 page 141, I compiled agent characteristics found in the literature; Table 25 uses them to characterise the roles we identified previously. Other methodologies, such as the one I reviewed in the state of the art, propose a strong formalisation of the roles not deemed necessary for our prototype; the priority of the project was to get a visible and demonstrable proof of concept and not a set of proven formal specifications.

	Ontology Archivist	Corporate Model Archivist	Annotation Archivist	Annotation Mediator	Directory Facilitator	Interface Controller	User Profile Manager	User Profile Archivist
Reactive				×		×		
Complex Mental State								
Graceful Degradation	×	×	×	×	×	×	×	×
Temporally continuity	×	×	×	×	×		×	×
Situated	×	×	×	×	×	×	×	×
Autonomy								
Individuality	×	×	×	×	×	×	×	×
Self-control	×	×	×	×	×	×	×	×
Flexible				×			×	
Goal-oriented				×			×	
Proactive				×			×	
Personality								
Communication (socially able)	×	×	×	×	×	×	×	×
Adaptability								
Customisable						×		
Learning							×	
Migration								
Mobility								
Visual representation						×		
Veracity	×	×	×	×	×	×	×	×
Benevolence	×	×	×	×	×	×	×	×
Rationality	×	×	×	×	×	×	×	×

Table 25 Complete role characteristics in CoMMA

For every role and for each property, it was decided whether the property was required by that role or not. The lines give the characteristics and the columns give the roles. The crosses × indicate that a given property (line) holds for a given role (column).

Discussing this table, Giovani Rimassa (from Parma University) proposed to summarise this analysis simplifying wherever possible. Adapting this to the last prototype, I summarised it here:

- No agent is mobile, migrates, has a complex mental state or a personality (at least according the definitions gave previously).
- No agent is purely reactive, but most of them are more reactive to social interactions than to the information environment except for the Interface Controller and the Annotation Mediator. The Interface Controller is 'sensing' its interface which could be considered as an essentially reactive

behaviour. The Annotation Mediator is sensing additions to annotation archives to notify such event to other agents.
- Each agent is benevolent, rational, respects veracity, communicates, degrades gracefully, enjoys individuality and self-control and is situated in an software environment.
- Apart from the Interface Controller, every agent is temporally continuous.
- The User Profile Manager requires learning capabilities.
- The Interface Controller requires customisation capabilities and a visual representation (GUI)

These considerations reduce Table 25 to Table 26.

		Ontology Archivist	Corporate Model Archivist	Annotation Archivist	Annotation Mediator	Directory Facilitator	Interface Controller	User Profile Manager	User Profile Archivist
Autonomy									
	Goal-oriented				×			×	
	Flexible				×			×	
	Proactive				×			×	

Table 26 Summarised role characteristics in CoMMA

The Annotation Mediator and the User Profile Manager are highly autonomous, whereas the Ontology Archivist, Corporate Model Archivist, Annotation Archivist, Directory Facilitator, Interface Controller and User Profile Archivist are weakly autonomous.

The Annotation Mediator and User Profile Manager, require an architecture supporting goal driven behaviour or at least knowledge-based mechanism and pro-activity.

The Ontology Archivist, the Corporate Model Archivist, the Annotation Archivist, the Directory Facilitator, Interface Controller and User Profile Archivists can use a simpler architecture with no need for explicit planning.

9.2.3 Social interactions

The sub-societies identification is followed by roles identification, but in parallel to roles specification, the social interactions are specified too as the `interactions` facet of the role card revealed. Likewise, some methodologies propose a strong formalisation of these interactions and for the same reason than for the roles, such formalisation was not deemed necessary for our first prototype.

Interactions consist of more than the sending of isolated messages and the conversation patterns need to be specified with protocols. Agents must follow these protocols for the MAS to work properly. Protocols are codes of correct behaviour, in a society, for agents to interact with one another. They describe a standard procedure to regulate information transmission between agents and they institutionalise patterns of communication occurring between identified roles. These institutional interactions sustain the social structures, appearing and disappearing with them.

To specify and document the interactions, we reused the AUML acquaintance graphs and protocol diagrams, a restriction of the UML collaboration diagrams [Bergenti and Poggi, 2000]. There are even more interactions than roles thus I shall not give all the details here; it would be useless for the purpose of describing the design and it is unnecessary to an overview of the architecture.

The definition of an interaction starts with an acquaintance graph at role level, that is a directed graph identifying communication pathways between agents playing the roles involved in an interaction scenario. A non directed edge denote that both agents playing the roles know each others. Example in Figure 71 shows an acquaintance diagram between the four roles Interface Controller, User Profile Manager, User Profile Archivist and Directory Facilitator before the login session interactions.

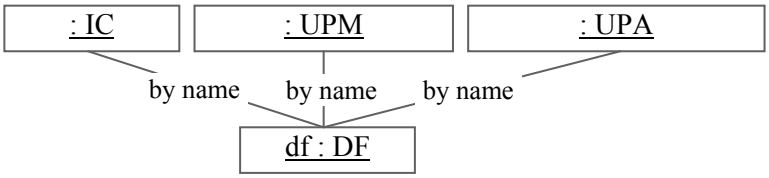


Figure 71 Acquaintance in User-dedicated Society before Login

This figure can be compared to the acquaintance diagram after the login session took place as depicted in Figure 72:

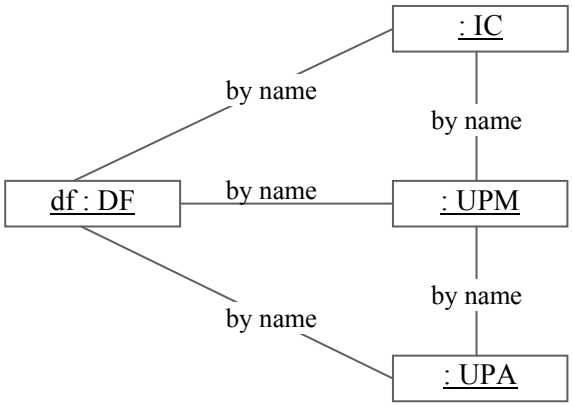


Figure 72 Acquaintance in User-dedicated Society after Login

We see from these diagrams that initially only the Directory Facilitator is known by its name; but then the login interactions required new acquaintances between the agent playing the Interface Controller role and the agent playing the User Profile Manager role and between the agent playing the User Profile Manager role and the agent playing the User Profile Archivist role.

To specify possible sequences of messages that can occur, we use the protocol diagram. In the example used previously concerning User Login, Figure 73 shows the sequence of messages that occurred omitting the failure cases which should normally be handled using alternative messages such as "2a: agree" / "2b: refuse" / "2c: not-understood".

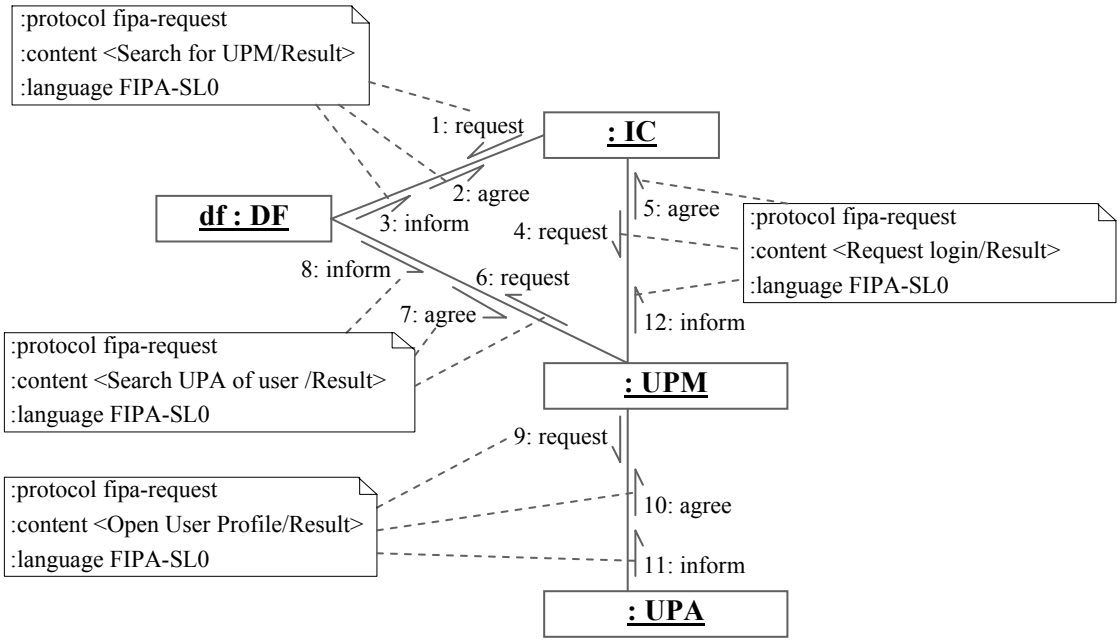
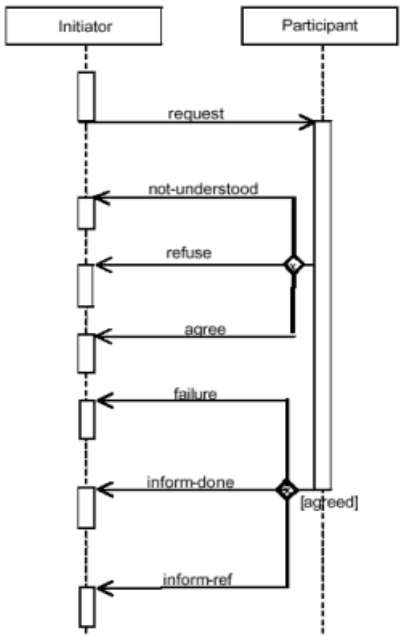


Figure 73 Login interactions an protocols

The acquaintance connections among the roles and the protocols adopted derive from both the organisational analysis and the use cases dictated by the application scenarios. Here in Figure 73, we can see an interaction based on four occurrences of the request protocol. Messages 1, 2 and 3 are the Request protocol initiated by the Interface Controller with the Directory Facilitator to get acquainted with the local User Profile Manager. Messages 4, 5 and 12 are the Request protocol initiated by the Interface Controller with the User Profile Manager to log its user into the system. Messages 6, 7 and 8 are the Request protocol initiated by the User Profile Manager with the Directory Facilitator to get acquainted with the User Profile Archivist that manages the profile of the user wanting to log in. And messages 9, 10 and 11 are the Request protocol initiated by the User Profile Manager with the User Profile Archivist to retrieve the profile and notify the it that the user is logged in.



These diagrams are not usable for large numbers of roles or long interactions since the number of possibilities and the messages rapidly make the diagram too clumsy and cluttered. However it is perfectly usable for focused interactions; accepting sometimes not to depict all the messages (e.g. Figure 73 omits failure cases). Generic protocols are depicted by FIPA using their interaction diagrams as shown in Figure 74. It is clear that for overall views of large sequences of interactions with multiple junctions in message sequences, other types of diagrams may be interesting such as Petri-nets and causal networks.

Figure 74 FIPA Request Protocol

For a given role, services and associated communication primitives may be documented using role diagrams [Bergenti and Poggi, 2000] that are directly linked to previous role cards. Figure 75 shows the performatives understood by the User Profile Archivist; more details can be added such as nested structure of the messages.

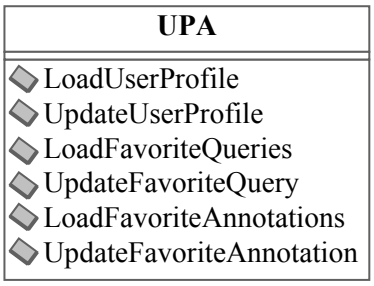


Figure 75 User Profile Archivist role diagram

The implementation of CoMMA relies on JADE [Bellifemine *et al.*, 2001], which is an open source MAS development platform compliant with the FIPA specifications. The agent communication language FIPA ACL is based, like its counterpart KQML [Finin *et al.*, 1994], on the speech act theory and comes with already standardised protocols to be used or extended with the semantics of their speech acts specified in the FIPA ontology.

In the first prototype, the content languages of the messages were (1) SL for the speech acts specified by FIPA for the envelop (Figure 76-a) and the speech acts involved in the CoMMA protocols (Figure 76-b); (2) RDF for the exchanged annotations and query patterns (Figure 76-c). The latest prototype uses RDF as an ACL.

```
a  QUERY-REF
   :sender(agent-identifier :name localUPM@apollo:1099/JADE)
   :receiver(set(agent-identifier :name AM@apollo:1099/JADE))
   :content
b    ((all ?x (is-answer-for
                 (query
      :pattern
c     <?xml version ="1.0"?> <rdf:RDF xml:lang="en"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:comma="http://www.inria.fr/acacia/comma#">
      <comma:Memo><comma:Designation>?</comma:Designation>
      </comma:Memo>
      </rdf:RDF>
b                ) ?x ) ) )
a  :reply-with  QuerylocalUPM987683105872
   :language  CoMMA-RDF
   :ontology  CoMMA-annotation-ontology
   :protocol  FIPA-Query
   :conversation-id  QuerylocalUPM987683105872 )
```

(a) FIPA ACL Envelop - (b) CoMMA SL0 query expression - (c) RDF Pattern

Figure 76 A message asking for the title of Memos

Messages are in FIPA ACL and use three ontologies:

- *FIPA ontology* providing the meaning of the speech acts used in the interaction protocols.
- *CoMMA management ontology* describing predicates such as "is-answer-for", "query" and slots such as ":pattern". It was written directly in the JADE format, but it is clear that eventually it should be merged with the O'CoMMA ontology in a branch providing primitives to describe the memory, its events and its tasks.
- *O'CoMMA* is used for the RDF query patterns and annotations.

9.2.4 Behaviour and technical competencies

From the role and interaction descriptions, the different partners of CoMMA proposed behaviours and implemented them in agent types (agent classes) that fulfil one or more roles. Interaction diagrams were also indicators of the sub-groups of designers that should interact because the behaviour they were in charge of were to play roles that would have interactions with one another.

The complete behaviour of an agent type combines behaviours implemented by the designers to accomplish the activities corresponding to the assigned roles. For instance, there is currently one agent type playing both the Ontology Archivist role and part of the Corporate Model Archivist role. Its behaviour contains both associated behaviours which are themselves made up of sub-behaviours handling the different tasks and interactions linked to these roles.

Behaviours are directly linked to the implementation choices and determine the responses, actions and reactions of the agent. The implementation of the behaviour is constrained by the role, but it is also subject to the toolbox of technical abilities available to the designers.

In the following, section we complete our top-down design with an overview of the agent types and behaviours implemented by the different partners of the CoMMA project.

9.3 Agents types and deployment

This section presents the implementation of the agent types and their integration. As shown in Figure 77 agent roles are not in a one-to-one relationship with agent types and each agent type required some specific technical ability to implement a behaviour fulfilling its attributed roles.

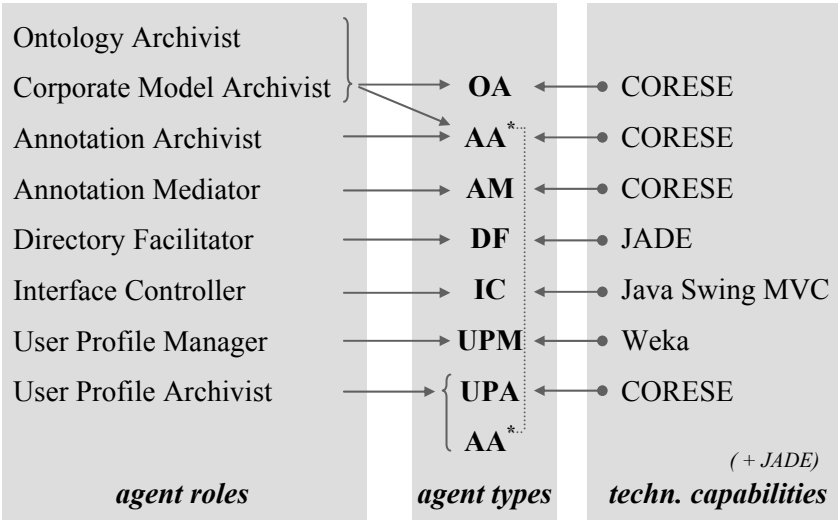


Figure 77 Allocating agent roles to agent types

This schema will be commented throughout the following sections briefly presenting the implemented agents and their use in a deployment.

9.3.1 Typology of implemented agents

9.3.1.1 Interface Controller Agent Class implementation

As shown in Figure 77, the Interface Controller role is played by one agent type: the IC agent. An IC agent is created at login and dies at logout. One of the major difficulty in implementing this agent is to try to provide a powerful and yet intelligible interface, that hides the complexity of the MAS and the semantic Web technologies.

The implementation of a relevant, user-friendly Graphic User Interface (GUI) has been one of the critical issues of the CoMMA project. The GUI can be seen as a window through which the user will be able to explore the CoMMA system functionalities. That does mean that the GUI is a key factor regarding the acceptance of the system by the end-users.

The end-users are not aware of XML/RDF and have limited skills in computer. Moreover, the RDF syntax is quite complicated. So the end-user must be able to handle the CoMMA system without knowing this syntax. RDF must not be visible to the user that will deal with concepts and properties through a graphical representation. The RDF syntax is generated by the GUI when needed. The mechanism used to display the queries or the annotations must offer a clear and immediate understanding of their meaning.

The necessity for the user to use the controlled vocabulary defined into the ontology and to browse large taxonomies can be too constraining. Following a "Z route" through the interface *i.e.,* top-left to top-right to bottom-left to bottom-right, we tried to make the selection of the relevant notions as natural and intuitive as possible:

- *top-left*: the GUI provides a terminological search to enter in the ontology. The user enters a keyword. A comparison is done with all the concepts, the properties and the possible synonyms defined into the ontology.
- *top-right*: the list of matching notions is proposed to the user who can select the most relevant one.
- *bottom-left*: selected notions are displayed and articulated; The GUI provides the user with a graph representation of the queries or annotations being built. Boxes are used for displaying concepts and lines represent the properties. By clicking on a box, the user can display all the information available on the related concept type (relations or attributes that can be instantiated appear in top-right) or on this specific instance (attribute values appear bottom-right).
- *bottom-right*: information on a specific concept instance are displayed in particular the attributes values.

As soon as the user has selected the entrance point within the ontology, the system is able to guide him during the whole process, by displaying the list of available concepts and properties that can be selected. The "Z route" process is iterative: first, the user selects a concept using a terminological search through the whole ontology; secondly, the user chooses among the properties that can be assigned to this concept; thirdly, the user can refine the new concept linked by this property; and so on, step by step, selecting alternatively a property or a concept, the user builds his own query or annotation from scratch or from an existing annotation or query saved in the user profile.

Queries allow attribute fields to be filled with variables or regular expressions. Annotation process starts with the statement of the URI of the annotated resource.

As a user profile is just a particular annotation about the user, it is managed as any other annotation. Thus, the user profile management environment is exactly the same as the annotation environment except that the user profile is sent to the User Profile Archivist and not to the Annotation Archivist to be saved by the system. Management or administrator rights were not implemented in the system since they were not one of our priority in proving the concept of a multi-agent-based memory management system.

Figure 78 shows a document annotation view which relates to an article in English which targets an organisation interested in knowledge management, multi-agent system, ontology and symbolic representation. All the properties of the "article" concept are available in the "related property" list box because it is the concept currently selected.

Figure 79 shows an example of a graphical query entered by the user. The user wants all the titles of reports that concern Cognitive Science and Multi Agent Technology. All the properties of the "report" concept are available in the "related property" list box because it is the concept currently selected.

This agent is implemented using: Java Swing, an XSLT engine and a micro Web-browser to display the processed XML and the results. It calls the user's HTML browser to display selected URL. The internal architecture in a Finite State Machine using compound behaviours in JADE; the behaviour is essentially event-driven.

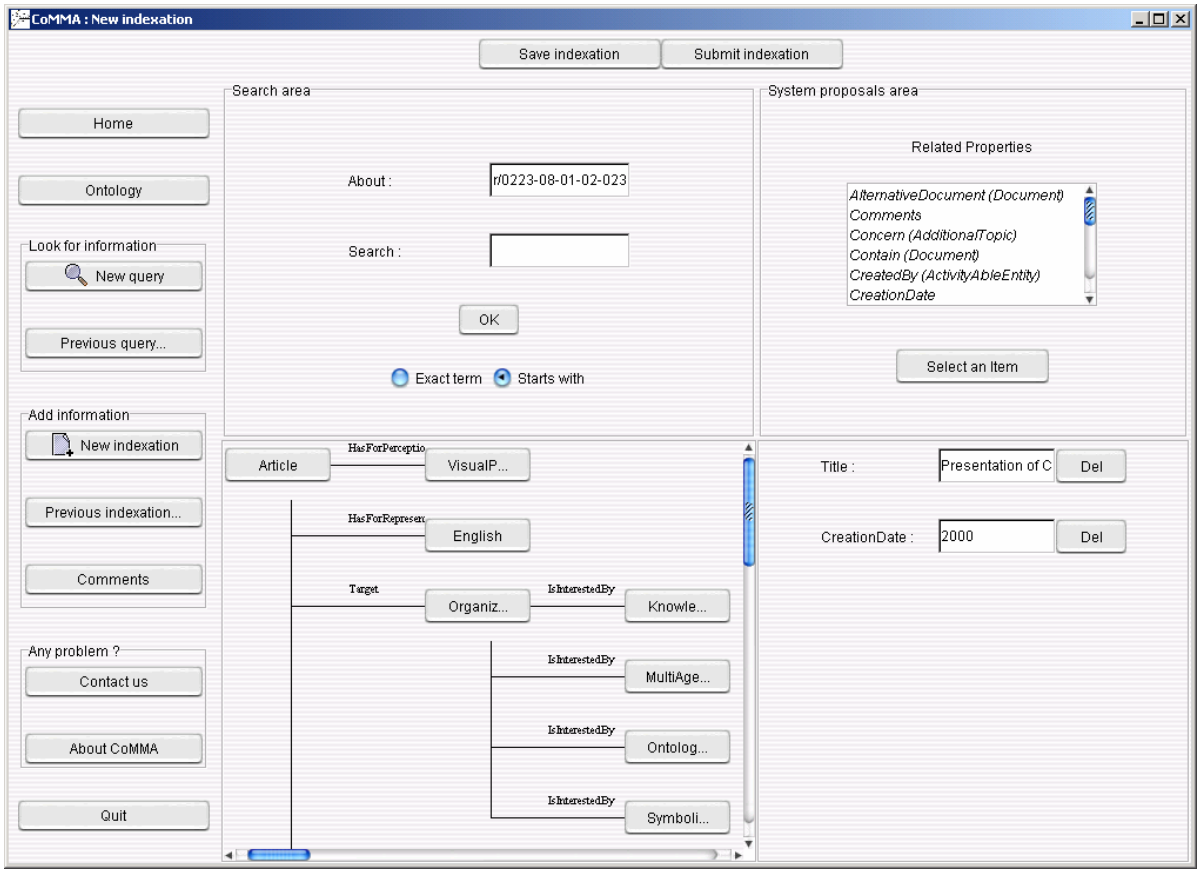


Figure 78 Creating an annotation

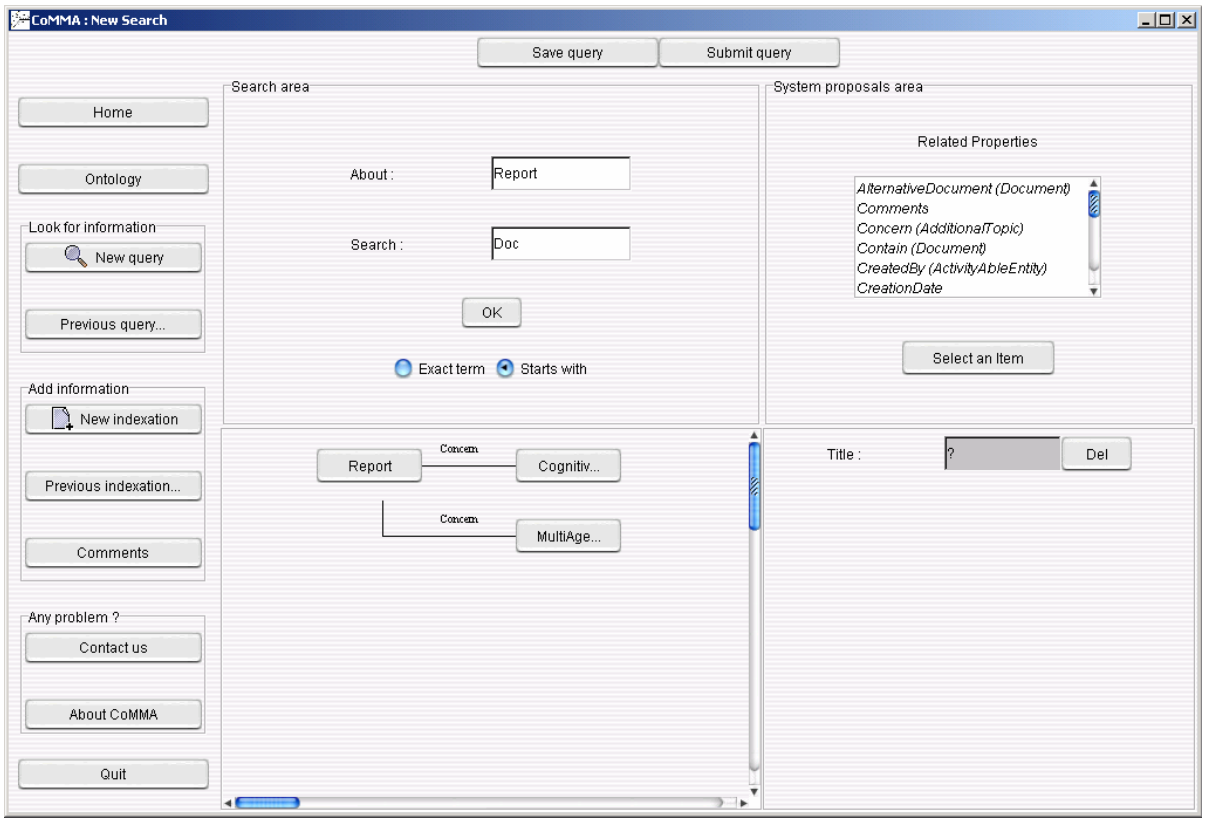


Figure 79 Formulating a query

9.3.1.2 User Profile Manager Agent Class implementation

As shown in Figure 77, the User Profile Manager role is played by one agent type: the UPM agent class. It uses machine learning techniques to learn the interest of the user and build an ordering relation to rank answers of the system and sort them before transmitting them to the Interface Controller. This agent was first implemented using the Weka library and then improved. The work on the machine learning techniques was carried out by our colleagues of the LIRMM as detailed in [Kiss and Quinqueton, 2001]; I shall only briefly explain the principle here.

Machine learning in the CoMMA system is devoted to user adaptation. The system prompts the user to get an opinion on the resources suggested after a push or a pull of information. The opinion is expressed as a rating and recorded. Then, the system extracts characteristic information from the documents (type and topics) and from the profile of users who participated in the ranking, in order to create generalisations to be applied in future document searches. This introduces a subjective relevance measure, that considers the user's profile to rank documents. In CoMMA the ontology provide the primitives to describe this context.

The machine learning component in the CoMMA system was designed to function both in the "pull" and "push" information retrieval scenarios. In the pull scenario, it acts as a mediator in the query processing cycle, using user and context dependent information to filter and sort the query results. While in the push subsystem, as soon as the confidence on ranking of documents is high enough, the same engine proposes the new documents to the user for which the predicted ranking passes above a certain threshold.

For the implementation of the system, several approaches were experimented. Most of these approaches have been proven to be inadequate due to reasons of flexibility, efficiency, large number of examples needed to function or because of the lack of explicit knowledge that was produced. Finally, the system was based upon a PART [Kiss and Quinqueton, 2001] method, for which the learning step is quick and produces logical rules, which are readable. This is extremely valuable as we are in a framework concerned of human and machine understandable information: it thus permits a formalisation of the learnt knowledge base in RDF.

The first testing and prototype were implemented using the state-of-the-art WEKA machine-learning package, as it was conform to all the choices done for the implementation of CoMMA. Then finally, we developed a specific Machine learning module based on the PART method [Kiss and Quinqueton, 2001], enhanced to take into account the specific characteristics of the description used in CoMMA.

The internal architecture exploits both Finite State Machines for protocols handling and machine learning to provide adaptation capabilities.

9.3.1.3 User Profile Archivist Agent Class implementation

As shown in Figure 77, one agent type, called UPA, is in charge of the part of the UPA role concerning storage and retrieval of user profiles for login and logout. Precise querying on user profiles is handled by another agent type (the Annotation Archivist) that belongs to the annotation-dedicated sub-society. This UPA agent is also in charge of the matchmaking of new annotations forwarded by the User Profile Manager; for this reason, it is implemented using CORESE API.

The internal architecture exploits both Finite State Machines for protocols and Knowledge-based systems (CORESE) for profile handling.

9.3.1.4 Directory Facilitator Agent Class implementation

CoMMA being based on the JADE platform [Bellifemine *et al.*, 2001], the agents of the connection sub-society are of two types defined by FIPA [FIPA]:

- Agent Management System (AMS) agent type: this agent of JADE is in charge of maintaining white pages where agents register themselves and ask for addresses of other agents on the basis of their name.
- Directory Facilitator (DF) agent type: this agent of JADE is in charge of maintaining yellow pages where agents register themselves and ask for addresses of other agents on the basis of a description of the services they can provide.

AMS and DF agents are implemented and delivered with the JADE platform. They are directly used by the other agent types developed in CoMMA to form the acquaintance needed for their roles. The current prototype only uses the agent type as a service description, but further service description refinement is done in a second step as it will be detailed for the annotation-dedicated sub-society.

The internal architecture exploits Finite State Machines.

9.3.1.5 Ontology Archivist Agent Class implementation

As shown in Figure 77, the roles from the sub-society dedicated to ontology and model were split over two agents:

- Ontology Archivist (OA) agent takes in charge the Ontology Archivist role and the part of the Corporate Model Archivist role in charge of managing and providing download of the organisational model in RDF.
- The Annotation Archivist (AA) agent from the annotation-dedicated society, takes in charge the participation of the annotations of the model to the solving of a query.

The AA agent will be detailed later. The Ontology Archivist needs to manipulate the ontology and is thus implemented using CORESE. The two main behaviours of the OA are responder behaviours to request for downloads of the ontology and the corporate model.

9.3.1.6 Annotation Mediator Agent Class implementation

As shown in Figure 77, the Annotation Mediator is implemented in one agent type: the AM agent class. It uses CORESE for the management of RDF(S) data. A complete description of this agent type and behaviour is given in the next chapter on distributed annotation management.

9.3.1.7 Annotation Archivist Agent Class implementation

As shown in Figure 77, the Annotation Archivist is implemented in one agent type: the AA agent class. It uses CORESE for the management of RDF(S) data. A complete description of this agent type and behaviour is given in the next chapter on distributed annotation management.

9.3.2 Deployment configuration

A configuration is an instantiation of an architecture with a chosen arrangement and an appropriate number of agents of each type. One given architecture can lead to several configurations. In the case of a multi-agent corporate memory system, the configuration depends on the topography and context of the place where the system is rolled out, thus it must adapt to this information landscape and change with it. The architecture has been designed so that the set of possible configurations covers the different corporate organisational layouts foreseeable. The configuration description is studied and documented at deployment time using adapted UML deployment diagrams to represent hosts (servers, front-end...), MAS platforms, agent instances and their acquaintance graph.

The deployment of the CoMMA MAS is constrained by the organisational structure of the company where it is installed. Moreover, the deployment is also driven by the network topology technical environment and its constraints (such as the topologies characteristics, the network maps and data rates, data servers location, gateways, firewalls, etc) and by interests area, where are the stakeholders (users, system managers, content providers…).

Even if the static model (agent roles) and the dynamic model (agent interactions) describe quite well the CoMMA MAS architecture, it is necessary for large deployment and maintenance to provide a third view that is the deployment model. This model can help a lot in understanding and managing an instance of the system; a highly distributed system deployed over a structured, managed corporate network may prove to be very complex.

Figure 80 shows a small example of what a deployment diagram may look like for the CoMMA multi-agent system.

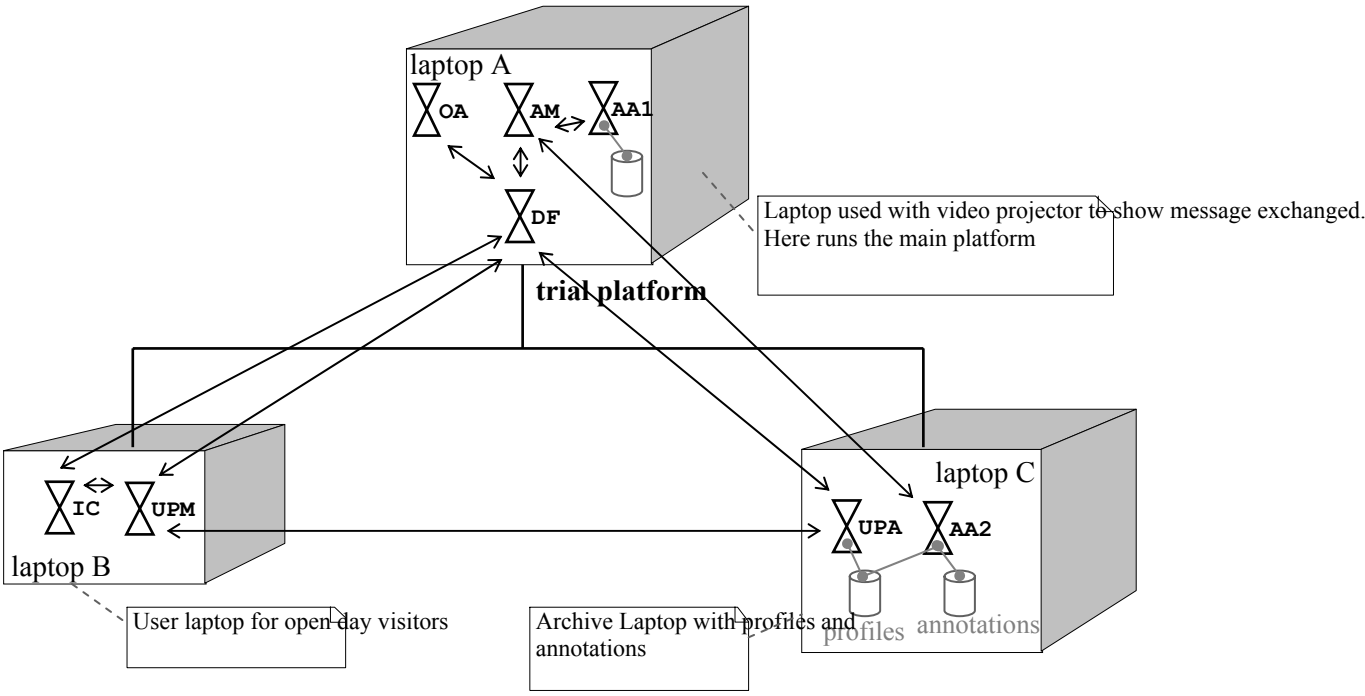


Figure 80 Deployment diagram of the open day

This deployment diagram corresponds to the configuration that was used for the Open Day demonstration using three laptops in network. I only depicted the most usual acquaintances (arrows). The configuration relies on one main platform running on laptop A; the other laptop only starts a container that connects to the platform in a transparent way.

9.4 Discussion and abstraction

Even if the description of the design was tedious, it showed each stage of the design rationale of a real experience. This section summarises the design process and abstracts the general characteristics of the approach from the perhaps too numerous details.

First it must be stressed that the approach taken here is only possible for a closed system *i.e.,* a system where the type of agents is fixed at design time. The system architecture is flexible enough to accept new roles for additional functionality with a minimum rework, but it was not designed for a full open world with rapidly changing configurations and agent types.

From the state of the art, the approach reused the general idea of a top-down functional analysis along the organisational dimension of the multi-agent system using the roles as a turning point between the macro-level of societal requirements and structures, and the micro-level of individual agents and their implementation. This is an organisational approach in the sense that the architecture is tackled, as in a human society, in terms of roles and relationships.

A MAS architecture is a structure that portrays the different families of agents and their relationships. A configuration is an instantiation of an architecture with a chosen arrangement and an appropriate number of agents of each type. One given architecture can lead to several configurations and a given configuration is tightly linked to the topography and context of the place where it is deployed; thus, the architecture must be designed so that the set of possible configurations covers the different layouts foreseeable.

The functional requirements of the system do not simply map to some agent functionality, but influence and are finally diluted in the dynamic social interactions of individual agents and in the set of abilities, roles and behaviours attached to them. Acknowledging this dilution, the architectural design starts from the highest level of abstraction (*i.e.,* the society) and by successive refinements (*i.e.,* nested sub-societies), it goes down to the point where agent roles and interactions can be identified. The architectural design comprises the following steps (Figure 81):

- Considering the functionalities requested for the system at the social level, we *identified dedicated sub-societies* of agents to handle the different general facets of these general functionalities.
- Considering each one of the sub-societies, we *identified a set of possible organisational structures* for them (in the present case: hierarchical, peer-to-peer, replication). Depending on the type of tasks to be performed, the size and complexity of the resources manipulated in each a sub-society, an organisational structures is preferred to another.
- From the organisational structure analysis, we *identified agent roles*, that is, the different positions an agent could occupy in a society and the responsibilities and activities assigned to this position and expected by others to be fulfilled. We studied the different role identified using role cards and comparing them along the agent characteristics usually identified in the literature.
- In parallel to role descriptions, we *identified interactions among agents* playing the roles. The role interactions are specified with protocols that the agents must follow for the MAS to work properly. The documentation of an interaction starts with an acquaintance graph at role level, that is a directed graph identifying communication pathways between agents playing the considered roles. Then, we specified the possible sequences of messages. The acquaintance network and the protocols are derived from the organisational analysis and the use cases dictated by the application scenarios.
- From the role and interaction descriptions, the different partners of CoMMA proposed and *implemented agent types* that fulfil one or more roles. *Behaviours* come from the implementation choices determining the responses, actions and reactions of the agent. The implementation of a behaviour is constrained by the associated role and interactions and is subject to the toolbox of

technical abilities available to the designers. Some roles were merged in one agent, some were split in two because part of the role corresponded to another existing role.

For a given instance of the architecture, the *configuration is studied and documented at deployment* time using deployment diagrams.

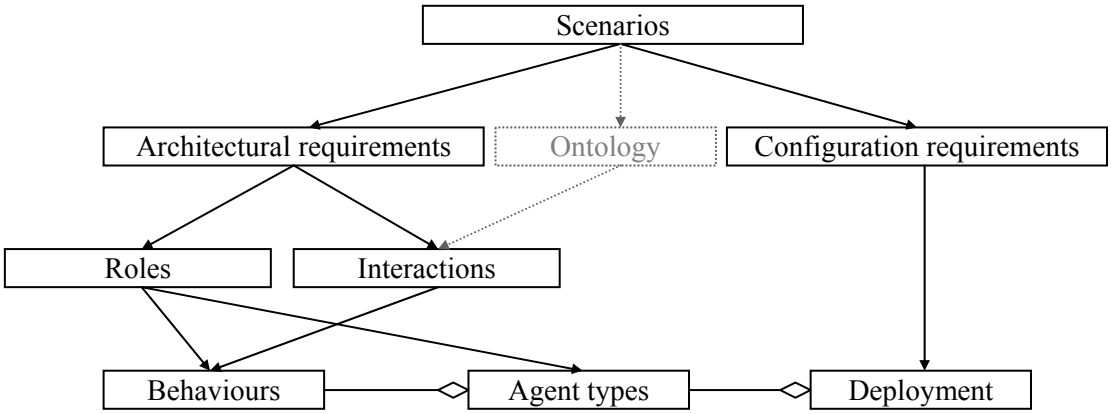


Figure 81 Top-down organisational and functional analysis

However what this chapter did show is that there is a strong need for design and documentation off-the-shelve tools to assist designers in their specification work, if we want the agent paradigm to make it to the real world as any other existing paradigm. These tools must really cover and assist design and deployment representation at each step of the lifecycle of a multi-agent system.

Finally, even if the architecture proposed here is designed for a closed system, and as I shall discuss in the perspectives, one can quite easily introduce new roles to add functionalities that will immediately benefit to the whole system.

10 Handling annotation distribution

I detail how some aspects of the conceptual foundations of the semantic web can support a multi-agent system in allocating and retrieving semantic annotations in a distributed corporate memory. I shall show how our agents exploit the underlying graph model, when deciding how to allocate new annotations and when resolving distributed queries. Three situations are presented:

- how agents allocate newly posted annotations: I shall explain how agents use the contract-net protocol together with a pseudo-distance between a new annotation and an archive of annotations for choosing the base where to store the annotation.
- how agents allocate the tasks involved in solving a query: I shall explain how agents use the nested query-ref protocol together with a description of the overlap between the query needs and the statistics over an archive of annotations for distributing sub-tasks in the form of sub-queries.
- how push is triggered: I shall briefly explain the mechanism triggering the push of newly submitted annotations to concerned users since it is done by the annotation mediator.

10.1 Issues of distribution

The duality of the definition of the word 'distribution' reveals two important problems to be addressed:

- Distribution means dispersion, that is the *spatial property of being scattered about*, over an area or a volume; the problem here is to handle the naturally distributed data, information or knowledge of the organisation.
- Distribution also means the act of 'distributing or *spreading* or apportioning'; the problem then is to make the relevant pieces of information go to the concerned agent (artificial or human). It is with both purposes in mind that we designed this sub-society.

My objective was to handle both aspects of the distribution of annotations over the Archivist Agents.

From distributed database systems [Özsu and Valduriez, 1999], we know that two types of fragmentation may be envisaged: horizontal and vertical. By drawing a parallel between data/schema and knowledge/ontology, we may say that in our case these two types of fragmentation would correspond to:

- *Horizontal fragmentation*: information is split according to the range of properties. For instance $site_1$ will store all the reports with a property 'title' ranging from "Alarming criminality in agent societies" to "Mass errors in behavioural control" and $site_2$ will store all the reports with a property 'title' ranging from "Naive implementation of resource distribution" to "Zeno paradox in iterative loops".
- *Vertical fragmentation*: information is split according to types of concepts and properties. For instance, $site_1$ will store all the reports with their titles and authors and $site_2$ will store all the articles with their abstracts and keywords.

Knowing this, the stake is to find mechanisms to decide *where to store newly submitted annotations* and *how to distribute a query* in order not to miss answers just because the needed information is split over several annotation archives. These two facets of distribution are linked since the performance of distributed query resolution is closely related to the choices made for the distribution of annotations.

Knowledge distribution is both the spatial property of knowledge of being scattered about and the act of diffusing knowledge. In a complete management cycle, the tasks of managing both facets of distribution to allocate storage and distribute exploitation are linked to one another in their performances by the strategies they adopt.

The two agent types involved in the management and exploitation of the distributed annotations are:

- the *Annotation Archivist* (AA): this agent is in charge of saving and querying the annotations of the corporate memory with its local resources. The Annotation Archivist is known by and communicates with its Annotation Mediator.
- the *Annotation Mediator* (AM): this agent is in charge of solving queries on the memory and allocating new annotations to the best suiting Annotation Archivist. The Annotation Mediator is thus in charge of getting in touch and managing the agents involved in the resolution process, in particular its AAs, and other AMs.

They form the annotation-dedicated society, in charge of handling annotations and queries in the distributed memory. Both queries and annotation submissions are generated by agents from the user-dedicated society and routed to the annotation-dedicated society. The latter is a hierarchical society: the agents playing the Annotation Mediator role (AM) are in charge of managing agents playing the

Annotation Archivist role (AA). The AM provides its services to other societies to solve their queries and, to do so, it requests the services of the AAs. On the other side, the AA role is attached to a local annotation repository and when it receives a request, it tries to fulfil it with its local resources in a fashion that enables the AM to handle the distributed dimension of the problem. The agents playing the role of AA and AM are benevolent and, once deployed, temporally continuous. Agents playing the AMs are in charge of handling query decomposition and allocation of new annotations to the AAs. We shall see how the ontology and the semantic web framework can support these agents in their tasks.

10.2 Differentiating archivist agents

In order to determine which AA should be involved during the solving of a query or to which one an annotation should be given, we compare the content of their archives thanks to a light structure called ABIS (Annotation Base Instances Statistics).

As shown in Table 27, the ABIS captures statistics, maintained by the AA, on the population of triples of its annotation base:
- The number of instances for each concept type; if a concept type of the ontology is missing in that list, it means there is no instance of that concept type in the annotation base.
- The number of instances for each property type; if a property type of the ontology is missing in that list, it means there is no instance of that property type in the annotation base.
- the number of instances for each family of properties (as explained below).

A family of properties is defined by a specialised signature corresponding to at least one instance present in the archivist base:

$$[\text{ConceptType}_x] \rightarrow (\text{PropertyType}_y) \rightarrow [\text{ConceptType}_z]$$

where the concept types are possibly more precise than the signature of $PropertyType_y$. For instance, if there exists a property *Author* with the following signature:

$$[\text{Document}] \rightarrow (\text{Author}) \rightarrow [\text{Person}]$$

we may have families of properties such as:

$$[\text{Article}] \rightarrow (\text{Author}) \rightarrow [\text{Student}]$$

$$[\text{Book}] \rightarrow (\text{Author}) \rightarrow [\text{Philosopher}]$$

This means that for each of these specialised signatures, there exists, in the archive of the corresponding AA, at least one instance using exactly these types. If a family does not appear in the ABIS, it means there is no instance of this very precise type.

Concept	Population
Report	57

...

Relation	Population
Concern	23
Firstname	89
Manage	69

...

Relation	Population	Upper Bound	Lower Bound
Manager → Manage → Worker	12		
Manager → Manage → Employee	57		
Employee → Firstname → Literal	89	Alain	Laurent
Article → Concern → Web	23		

...

Table 27 ABIS (Annotation Base Instances Statistics)

The ABIS is built when annotations are loaded by the Annotation Archivist; the agent decomposes the corresponding Conceptual Graphs generated in CORESE into binary relations and identifies the possible isolated vertices to include them in the statistics. For literal properties, the bounding interval $[B_{low}, B_{up}]$ of their literal values is calculated. The ABIS is updated each time an annotation is loaded in the base and after applying rules, transitivity and symmetry completion to the base. The only problem that remains in the current prototype is to take into account reflexive properties. The reflexivity is calculated dynamically during the projection process. Therefore, it is not taken into account by basic statistics on the base content, however this should not disturb the algorithm too much and can therefore, be overlooked at least for the CoMMA prototype.

To reuse the terminology of distributed databases, the ABIS structure captures information about the vertical and horizontal contribution of an Annotation Archivist to the overall distribution. The ABIS captures the types of concepts and relations for which an AA contributes to the memory. It does not capture the structural information of the annotations, but it is enough to get an idea of the type of knowledge an archive keeps for the memory; it is a way to compare the specialisation of the Archivist Agents in terms of the content of their base.

As shown in Figure 82, the ABIS structure can be marshalled into an XML document using tags from the namespace of CORESE and from RDF(S).

```
1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <cos:ArchiveStatistics xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cos="http://www.inria.fr/acacia/corese#">
3
4   <rdfs:Class rdf:about="http://www.inria.fr/acacia/comma#Report">
5    <cos:NumberOfInstances>57</cos:NumberOfInstances>
6   </rdfs:Class>
7
8   <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Manage">
9    <cos:NumberOfInstances>12</cos:NumberOfInstances>
10   <cos:RangeType>http://www.inria.fr/acacia/comma#Manager</cos:RangeType>
11   <cos:DomainType>http://www.inria.fr/acacia/comma#Worker</cos:DomainType>
12  </rdf:Property>
13
14  <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Manage">
15   <cos:NumberOfInstances>57</cos:NumberOfInstances>
16   <cos:RangeType>http://www.inria.fr/acacia/comma#Manager</cos:RangeType>
17   <cos:DomainType>http://www.inria.fr/acacia/comma#Employee</cos:DomainType>
18  </rdf:Property>
19
20  <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Firstname">
21   <cos:NumberOfInstances>89</cos:NumberOfInstances>
22   <cos:RangeType>http://www.inria.fr/acacia/comma#Employee</cos:RangeType>
23   <cos:DomainType>http://www.w3.org/2000/01/rdf-schema#Literal</cos:DomainType>
24   <cos:UpperValue>Alain</cos:UpperValue>
25   <cos:LowerValue>Laurent</cos:LowerValue>
26  </rdf:Property>
27
28  <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Concern">
29   <cos:NumberOfInstances>23</cos:NumberOfInstances>
30   <cos:RangeType>http://www.inria.fr/acacia/comma#Manager</cos:RangeType>
31   <cos:DomainType>http://www.inria.fr/acacia/comma#Employee</cos:DomainType>
32  </rdf:Property>
33
34  </cos:ArchiveStatistics>
```

Figure 82 ABIS marshalled structure

The statistics on concepts are given in the marshalled structure because they cannot be derived from the rest of the statistics: a concept could be alone (isolated vertex in conceptual graphs) in an annotation (e.g.: `<document rdf:about="..."/>`) and therefore, not appearing in the property signatures.

Property signatures are not marshalled since they are easily derived from the statistics on the families of properties. For instance in the above example of Table 27, the property 'manage' has 69 instances that is the sum of the two populations appearing in the property instances table (12 and 57).

When a system is deployed, Annotation Archivists are started, but they may have no annotation in their bases. Their statistics being void, the ABIS is not relevant to compare their bids. Moreover, when deploying the system, it is interesting to be able to specialise individual agents according to the topography of the company network (e.g. an AA on a machine of Human Resources department for users' profile, another for scientific documents in each research group, etc.).

As shown in Table 28, the CAP (Card of Archives Preferences) is a light structure that captures the RDF properties for which the agent has a preference and, if specified, their range boundaries. Any specialisation of these properties is then considered to be part of the preferences of the AA. The CAP can be used to initiate a base and keep a trace of its initial specialisation. These fragmentation choices are made by the administrators when deploying the agents.

Relation	Upper Bound	Lower Bound
Report → Title → Literal	A	L
Article → Abstract → Literal	A	Z
Article → Concern → Computer Science		
Page Web → Concern → Computer Science		
Person → Name → Literal	M	R

Table 28 Card of Archive Preferences

If needed, this structure could be tuned with, for instance, 'population offsets' giving the degree of affinity of an Annotation Archivist for a given type of properties.

10.3 Annotation allocation

In this section, I present how agents allocate newly posted annotations. An important hypothesis is that agents do not break up annotations, they store them as one block. I shall explain how agents use the contract-net protocol [Davis and Smith, 1983] together with a pseudo-distance between a new annotation and an archive of annotations for choosing the base where to store the annotation.

10.3.1 Allocation protocol

Agents involved in the allocation of the new annotation are:
- the Interface Controller, through which the annotation is built and submitted.
- the User Profile Manager, that observes and forwards the annotation.
- the Annotation Mediator, that manages the allocation process.
- the Annotation Archivist, that manages local annotation archives.

The Interface Controller, and the User Profile Manager are essentially involved in the emission of the new annotation and I shall not consider them any further here.

Although the agents considered in CoMMA are benevolent and fully collaborating, the lack of individual interest does not mean that a situation like a market or a negotiation cannot be imagined. Indeed the problem of allocation of an annotation can be seen as a market where each Archivist Agent offers its services and where the Annotation Mediator in charge of allocating a new annotation is looking for the best 'contract'.

There exists at least two options during the evaluation process:
- The Annotation Mediators send the newly posted annotation to the Annotation Archivists, and the Annotation Archivists send back a bid; the best one gains the bid.
- The Annotation Archivists send their ABIS and CAP to the Annotation Mediators and the Annotation Mediators calculate the appropriateness of each Annotation Archivist when a new annotation is posted: the Annotation Mediators then proceed to contact the best Annotation Archivist.

There are pros and cons for each option:
- In the first option, the Archivist Agents must be trustworthy, must use the same calculation method for bids, but agents do not need to exchange ABIS and CAP each time they change and only the result of the bid calculation will be exchanged.
- In the second option, the Annotation Mediators ensure a fair and exact evaluation of the relevancy of the Annotation Archivists with their own measure, but the ABIS and CAP will have to be exchanged and the bid calculation will be centralised.

CoMMA agents are benevolent, non-antagonistic and fully co-operative due to the fact that they deal with a corporate semantic intraweb memory. Therefore, the first option seemed to be the best:
- The Annotation Mediators deal with the comparison of the different bids of the Annotation Archivists and grant the annotation to the best one.
- The Annotation Archivists evaluate the similarity between the newly posted annotation and their base and send back this evaluation to the Annotation Mediators.

Remark: I made an important simplifying hypothesis: the submitted annotations are not broken down *i.e.,* we store them as one block.

As shown in Figure 83, the protocol is a nested Contract-net with two levels: local AM → local AAs & [other AMs & → local AAs]. The diagram corresponds to the following scenario description:

1. The Interface Controller builds a well-formed and valid annotation using the O'CoMMA ontology.
2. The Interface Controller sends the request for adding the annotation to the User Profile Manager it is acquainted with.
3. The User Profile Manager contacts an Annotation Mediator. We call it the localAM, it is the first one found by the Directory Facilitator, but it could be chosen on more complex criteria such as the workload.
4. The LocalAM calls upon the Directory Facilitator to look for other Annotation Mediators.
5. The LocalAM sends a call for proposal to all the other Annotation Mediator.
6. The LocalAM and the other Annotation Mediators send a call for proposal to their Annotation Archivists.
7. Each Annotation Archivist sends back the result of its bid calculation function to its Annotation Mediator.
8. The LocalAM and the other Annotation Mediators choose the best candidate among their Annotation Archivists and send a refusal to the others.
9. The other Annotation Mediators send the result of their best candidate to the LocalAM.
10. The LocalAM compares results and grants the annotation to the best proposal and informs the others that their proposals are refused.
11. The Annotation Mediators losers forward refusal to their best Annotation Archivist.
12. If the best proposal is an other Annotation Mediator, then this Annotation Mediator grants the annotation to its best Annotation Archivist, otherwise it is directly granted to the best Annotation Archivist of the LocalAM.
13. The Annotation Archivist winner adds the annotation to its base.
14. The Annotation Archivist winner informs its Annotation Mediator of the addition, if need be the AM informs the LocalAM of the addition.
15. The LocalAM forwards acceptation to the User Profile Manager.
16. The User Profile Manager forwards acceptation to the Interface Controller.

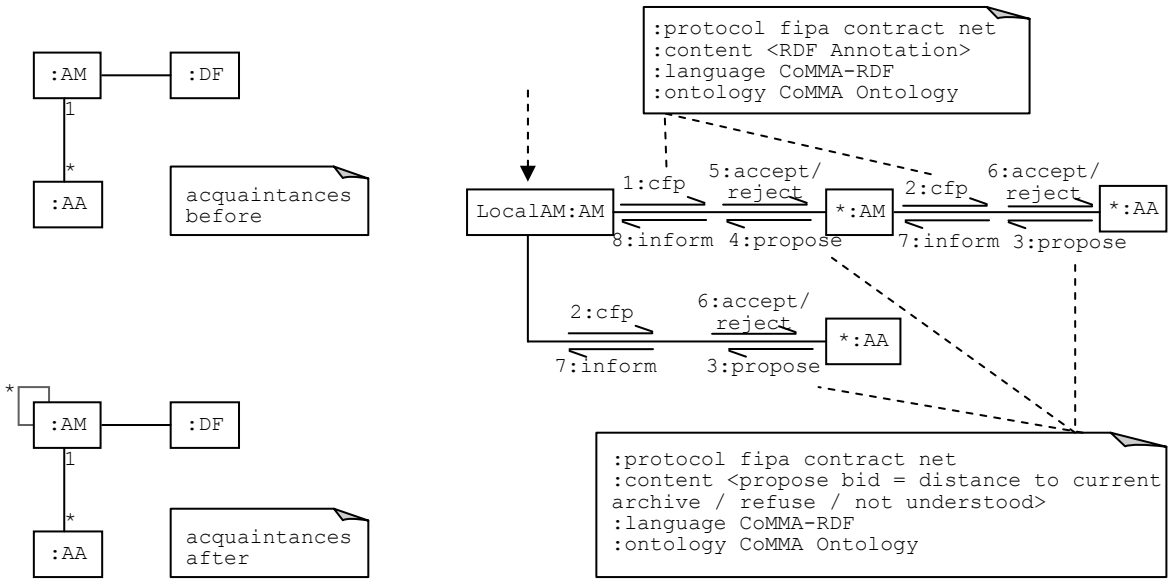


Figure 83 Acquaintance and protocol diagrams in annotation allocation

The whole process relies on the ability to bid and compare bids. This will be detailed in the following sections.

10.3.2 Pseudo semantic distance

In order to allocate a newly posted annotation, we need to measure how close it is from an Annotation Archivist ABIS and CAP to decide which Annotation Archivist should win the bid. Therefore, we need to measure the distance between an annotation and a base and to compare the different distances measured.

The calculation of the measure *i.e.,* a utility/adequacy function or a distance can take into account: the CAP, the ABIS, the resources available on the machine (CPU, hard disk space, etc.), the number of annotations already archived, some randomisation in order to ensure uniform distribution, etc.

I present thereafter the different steps to calculate the distance we actually use in the prototype.

10.3.2.1 Definition of constants

I define here the constants at play in the formula proposed for the pseudo-distance.

$$\mathrm{Max_L} = 256 \tag{1}$$

is the maximum range for an ASCII byte code of a character.

$$\mathrm{Max_C} \tag{2}$$

is the maximum path length in the subsumption hierarchy of the primitive concept types from the root to a leaf. It is calculated by recursive exploration of the subsumption hierarchy and it is usually called the depth of the ontology.

$$\mathrm{Max_R} \tag{3}$$

idem for primitive conceptual relation types.

$$\mathrm{N} = \mathrm{Max_C}*2/\ \mathrm{Max_L} \tag{4}$$

is the constant used to normalise the distances when combining distances on literals and distances on primitive types.

$$\mathrm{W_C}=4 \tag{5}$$

$$\mathrm{W_R}=8 \tag{6}$$

$$\mathrm{W_L}=1 \tag{7}$$

are weights respectively for concept types, conceptual relation types and literals. They are used to balance the importance of these factors in the pseudo-distance calculations.

10.3.2.2 Distance between two literals

Some properties have a literal range type that we need to compare. Let $C_{X,i}$ the ASCII byte code of the i^{th} character in upper case ($C_{X,i} \in [0,\mathrm{Max_L}[$) of $Lit_X=C_{X,0},C_{X,1},C_{X,2},C_{X,3},...,C_{X,s}$ a literal coded by the sequence of length $s+1$ of the codes of its characters. Let:

$$\mathrm{Abscissa}(\mathrm{Lit_X}) = \sum_{i=0..s} \frac{C_{\mathrm{x,i}}}{\mathrm{Max_L}^{i}} \tag{8}$$

This Abscissa is positive or null:

$$\mathrm{Abscissa}(\mathrm{Lit_X}) \geq \sum_{i=0..l} \frac{0}{\mathrm{Max_L}^{i}} = 0 \tag{9}$$

The Abscissa is bounded by *B* with:

$$\mathrm{B} = \sum_{i=0..l} \frac{\mathrm{Max_L} - 1}{\mathrm{Max_L}^{i}} = (\mathrm{Max_L} - 1) \cdot \sum_{i=0..l} \frac{1}{\mathrm{Max_L}^{i}} \tag{10}$$

We recognise the sum of a finite geometric sequence.

$$B = (\mathrm{Max_L} - 1) \cdot \frac{1 - \left(\frac{1}{\mathrm{Max_L}}\right)^{l+1}}{1 - \left(\frac{1}{\mathrm{Max_L}}\right)} = \mathrm{Max_L} - \frac{1}{\mathrm{Max_L}^{l}} < \mathrm{Max_L} \tag{11}$$

Thus $Abscissa(Lit_X) \in [0, Max_L[$, which explains the value of *N*.

We now consider the difference:

$$D = \mathrm{Abscissa(Lit_B)} - \mathrm{Abscissa(Lit_A)} \tag{12}$$

where Lit_A and Lit_B are two literals. If they are the same, their abscissas are equal and $D=0$. Else let Lit_A come before Lit_B in alphabetical order. This means that the first difference in reading these strings is a character in Lit_A that comes alphabetically before the character at the same position in Lit_B. This can be formalised as:

> If $Lit_A < Lit_B$ then $\exists i \in [0..s]$ such that $\forall j < i \; C_{A,j} = C_{B,j}$ (*i.e.,* the strings may have a common beginning) and $C_{A,i} < C_{B,i}$ *i.e.,* the first characters that are different, follow the same alphabetical order as the two strings they belong to.

The value of *D* (complementing the shortest string with characters of code 0 if necessary, so that both strings have the same length) is given by:

$$D = \sum_{\mathrm{k=0..s}} \frac{C_{B,k} - C_{A,k}}{\mathrm{Max_L}^{k}} = \frac{C_{B,i} - C_{A,i}}{\mathrm{Max_L}^{i}} + \sum_{\mathrm{k=i+1..s}} \frac{C_{B,k} - C_{A,k}}{\mathrm{Max_L}^{k}} \tag{13}$$

The absolute value of the remaining sigma sum is bounded by:

$$B' = \sum_{\mathrm{k=i+1..s}} \frac{\mathrm{Max_L} - 1}{\mathrm{Max_L}^{k}} = \frac{\mathrm{Max_L} - 1}{\mathrm{Max_L}^{i+1}} \cdot \sum_{\mathrm{k=0..s-i-1}} \frac{1}{\mathrm{Max_L}^{k}} \tag{14}$$

Here again we recognise the sum of a finite geometric sequence:

$$B' = \frac{\mathrm{Max_L} - 1}{\mathrm{Max_L}^{i+1}} \cdot \frac{1 - \left(\frac{1}{\mathrm{Max_L}}\right)^{s-i}}{1 - \left(\frac{1}{\mathrm{Max_L}}\right)} = \frac{1 - \left(\frac{1}{\mathrm{Max_L}}\right)^{s-i}}{\mathrm{Max_L}^{i}} < \frac{1}{\mathrm{Max_L}^{i}} \tag{15}$$

Therefore, as $C_{A,i} < C_{B,i}$ we proved that:

$$\frac{C_{B,i} - C_{A,i}}{\mathrm{Max_L}^{i}} \geq \frac{1}{\mathrm{Max_L}^{i}} > \left| \sum_{\mathrm{k=i+1..s}} \frac{C_{B,k} - C_{A,k}}{\mathrm{Max_L}^{k}} \right| \tag{16}$$

And we can conclude that $D>0$, which means that

$$\mathrm{Lit_A} < \mathrm{Lit_B} \Leftrightarrow \mathrm{Abscissa(Lit_A)} < \mathrm{Abscissa(Lit_B)} \tag{17}$$

Based on the abscissa we define an Euclidean distance:

$$Dist_L(Lit_A, Lit_B) = | Abscissa(Lit_B) - Abscissa(Lit_A) | \quad (18)$$

To avoid imprecision problem, the distance is implemented following the mathematically equivalent formula (19) instead of (18) where rounding of abscissa calculation is cumulated.

$$Dist_L(Lit_A, Lit_B) = \left| \sum_{i=0..s} \frac{C_{B,i} - C_{A,i}}{Max_L^{\,i}} \right| \quad (19)$$

As an example, if Lit_A = *"abandon"* and Lit_B = *"accent"* then:
- $Abscissa(Lit_A) \sim 65.25880898635597$
- $Abscissa(Lit_B) \sim 65.26274521982486$
- $Dist_L(Lit_A,Lit_B) \sim 0.0039362334688917144$

This distance is currently used in the system, but since a lexicographical distance cannot be qualified of a semantic distance, the overall distance used for bidding is only pseudo-semantic, and we shall see in the perspectives what can be envisaged to improve that point.

10.3.2.3 Pseudo-distance literal - literal interval

The ABIS and CAP provide bounding interval for literal properties. We define a pseudo-distance between a literal value Lit_X from an annotation and a range $[B_{low}, B_{up}]$ from those structures:

$$Dist_I(Lit_X, [B_{low}, B_{up}]) = 0 \text{ if } Lit_X \in [B_{low}, B_{up}] \text{ else } = Min(Dist_L(Lit_X, B_{low}), Dist_L(Lit_X, B_{up})) \quad (20)$$

This is only a pseudo-distance since it is not an application from Literal×Literal to $\Re^+$, but from Literal×[Literal,Literal] to $\Re^+$. For this reason the overall distance used for bidding is only a pseudo-distance and likewise we shall see in the perspectives what can be envisaged to improve that point.

10.3.2.4 Distance between two ontological types

To compare two primitive types, we use the length, in number of subsumption links, of the shortest path between these types in the hierarchies of their supertypes. The calculation of this distance is a problem equivalent to searching the least common supertype and the two distances from this supertype to the considered types:

$$Dist_H(T_1,T_2) = SubPath(T_1,LCST) + SubPath(T_2,LCST) \quad (21)$$

where *LCST* is the Least Common Super Type of T_1 and T_2 and *SubPath(T,ST)* is the length, in edges, of the subsumption path from a type *T* to one of its supertypes *ST*. *LCST* is calculated by a recursive synchronised exploration of the ancestors of T_1 and T_2.

This distance measures a semantic closeness since the least common supertype of two types captures what these types have in common.

$Dist_H$ complies to the four features of distances:

First: the distance from a type to itself is null.

$$Dist_H(T_1,T_1) = 0 \quad (22)$$

Proof: the least common super-type of (T_1,T_1) is T_1 and the shortest path to go from a node to itself is not to cross an arc.

Second: the distance is symmetric.

$$Dist_H(T_1,T_2) = Dist_H(T_2,T_1) \quad (23)$$

Proof: the considered path is not directed and therefore, the shortest path from T_1 to T_2 is also the shortest path from T_2 to T_1.

Third: a null distance can only be obtained if the two types are merged:

$$Dist_H(T_1,T_2) = 0 \Rightarrow T_1=T_2 \quad (24)$$

Proof: since the only way to have a null distance is not to cross an arc, it is only possible if the two nodes at the extremity of the path are merged.

Fourth: the distance between two points is lesser than or equal to the distance of a path going through a third type.

$$Dist_H(T_1,T_3) \leq Dist_H(T_1,T_2) + Dist_H(T_2,T_3) \quad (25)$$

Proof: this is proved *ad absurdio*: if $Dist_H(T_1,T_2) + Dist_H(T_2,T_3)$ was smaller than $Dist_H(T_1,T_3)$, this would mean that there exists a path $(T_1,...,T_2,...,T_3)$ shorter than the shortest path from T_1 to T_3, which is absurd. Therefore, $Dist_H$ is a distance.

To compare two concept types or two relation types, we use the length (in number of edges or 'hops') of the shortest path between these types going through a least common super type in the hierarchical graph of the ontology; this problem is close to the calculation of the smallest common subsumer.

A simple algorithm is used for the calculation and it is described in Figure 84:

```
1   // Declaration
2   Let Type₁ and Type₂ be the types in parameters of the distance
3   Let Ancestors₁  := ∅ and Ancestors₂  := ∅ be sets of couples (Type, Height)
4
5   // trivial case
6   If Type₁ ≡ Type₂ return 0
7
8   // Ancestor calculation
9   Let Parents₁ and Parents₂ and GrandParents be temporary sets of types
10  Let Height := 0 be a temporary integer
11  Parents₁ := { Type₁ }
12  Parents₂ := { Type₂ }
13  While (Parents₁ ≠ ∅) and (Parents₂ ≠ ∅)
14   {
15    GrandParents := ∅
16    For each P ∈ Parents₁
17     {
18      Ancestors₁ := Ancestors₁ ∪ { (P, Height) }
19      GrandParents := GrandParents ∪ DirectSubsumersOf(P)
20     }
21    Parents₁ := GrandParents
22    GrandParents := ∅
23    For each P ∈ Parents₂
24     {
25      Ancestors₂ := Ancestors₂ ∪ { (P, Height) }
26      GrandParents := GrandParents ∪ DirectSubsumersOf(P)
27     }
28    Parents₂ := GrandParents
29    Height := Height + 1
30   }
31
32  // Distance Calculation
33  Let Distance := 2 * OntologyDepth + 1 be a temporary integer
34  For each (Type₁, Height₁) ∈ Ancestors₁
35    For each (Type₁, Height₂) ∈ Ancestors₂
36      if Height₁+Height₂ < Distance then Distance := Height₁+Height₂
37  return Distance
```

Figure 84 Algorithm of the concept and relation types distance

The idea is to go up the hierarchy in parallel and find the ancestors with their distance and then find the LCST through which the path is the shortest. The algorithm takes into account multiple inheritance. If the hierarchy is not a connected graph, then it may happen that for two given concept types, there exists no path between them (it means that the hierarchies stop at different roots without having found any common ancestor). In that case, we use the upper bound of the length equal to (MaxDepthConceptHierarchy *2+1) for the concepts or (MaxDepthRelationHierarchy *2+1) for the relations.

10.3.2.5 Distance between a concept type and a literal

I decided that the distance between a primitive type and an arbitrary literal is a constant greater than any type distance. Let

$$Dist_{LC}(T_1,L_X) = (Max_C \times 2 + 1) \quad (26)$$

If one prefers to consider a literal as a basic type at the top of the hierarchy, then we could replace (26) by (27):

$$Dist_{LC}(T_1,Lit_X) = Depth(T_1) + 1 \quad (27)$$

where $Depth(T_1)$ is the length of the path from the root of the hierarchy to the primitive type T_1.

10.3.2.6 Pseudo-distance annotation triple - property family

Let $Triple_A = (T_{RA}, T_{A1}, T_{A2})$ a triple from an annotation and let $Triple_B = (T_{RB}, T_{B1}, T_{B2})$ a triple from the ABIS.

In an RDF triple, T_{A1} and T_{B1} are primitive concept types, let

$$D_{C1} = W_C \times Dist_H (T_{A1}, T_{B1}) \quad (28)$$

Now, considering the types T_{A2} and T_{B2}:

- If both are primitive concept types then let:

$$D_{C2} = W_C \times Dist_H (T_{A2}, T_{B2}) \quad (29)$$

- If one is a primitive concept type T and the other is a literal L

$$D_{C2} = W_C \times Dist_{LC}(T,L) \quad (30)$$

- If both types are literals then from the ABIS we know $[B_{low}, B_{up}]$ and from the annotation we know the literal Lit_X. Let:

$$D_{C2} = W_L \times N \times Dist_I(Lit_X, [B_{low}, B_{up}]) \quad (31)$$

Finally, we calculate the distance between the relation types, let

$$D_R = W_R \times Dist_H(T_{RA}, T_{RB}) \quad (32)$$

The final pseudo-distance between the annotation triple and a property family of the ABIS is given by:

$$Dist_{TFABIS}(Triple_A, Triple_B) = D_{C1} + D_R + D_{C2} \quad (33)$$

If D_{C1} and D_{C2} were real distances then $Dist_{TFABIS}$ would be a Manhattan distance.

10.3.2.7 Pseudo-distance annotation triple - ABIS

The pseudo-distance between a triple and an ABIS is the minimal pseudo-distance between this triple and the ABIS triples.

$$Dist_{TABIS}(Triple,ABIS) = \min_{Triple_i \in ABIS}(Dist_{TFABIS}(Triple,Triple_i)) \quad (34)$$

10.3.2.8 Pseudo-distance annotation triple - CAP

The calculation of the pseudo-distance $Dist_{TCAP}(Triple,CAP)$ is the same as for the ABIS except for the primitive type distance: when comparing two triples, if the type of the annotation is a specialisation of the type of the triple from the CAP, the length of the path between them is set to *0*. This is to take into account the fact that the CAP captures preferences and that anything more precise (as a specialisation) is included in the preferences.

At this point we can represent our structure summarising the information needed to evaluate the distance of an annotation from an archivist as a vector:

	0
Distance to CAP	8,34
	2,25
	0
	2
Distance to ABIS	2,5
	1,3
	1

This structure could be sent by the Annotation Archivist to the Annotation Mediator to make the final comparison for instance. The final operator (sum of the distance, maximum, minimum, etc.) could be chosen by the Annotation Mediator. It could also be sent together with other criteria (CPU load, size of the archive, etc.)

In the current prototype, the calculation is done by the Annotation Archivist using a simple sum as described in the following sections.

10.3.2.9 Pseudo-distance annotation - ABIS

We sum the pseudo-distances for the triples of the annotation:

$$\mathrm{Dist}_{\mathrm{AABIS}}(\mathrm{An}_{\mathrm{X}}, \mathrm{ABIS}) = \sum_{Triple_j \in An_X} \mathrm{Dist}_{\mathrm{TABIS}}(\mathrm{Triple}_{\mathrm{j}}, \mathrm{ABIS}) \quad (35)$$

where An_X is an annotation and ABIS is the ABIS of an AA.

10.3.2.10 Pseudo-distance annotation - CAP

We sum the pseudo-distances for the triples of the annotation:

$$\mathrm{Dist}_{\mathrm{ACAP}}(\mathrm{An}_{\mathrm{X}}, \mathrm{CAP}) = \sum_{Triple_j \in An_X} \mathrm{Dist}_{\mathrm{TCAP}}(\mathrm{Triple}_{\mathrm{j}}, \mathrm{CAP}) \quad (36)$$

where An_X is an annotation and CAP is the CAP of an AA.

10.3.2.11 Pseudo-distance annotation - AA

We sum the pseudo-distances to ABIS and CAP:

$$\mathrm{Dist}(\mathrm{An}_X, \mathrm{AA}_y) = \mathrm{Dist}_{\mathrm{AABIS}}(\mathrm{An}_X, \mathrm{ABIS}_y) + \mathrm{Dist}_{\mathrm{ACAP}}(\mathrm{An}_X, \mathrm{CAP}_y) \quad (37)$$

where AA_y is an archivist agent, An_X is an annotation and $ABIS_y$ and CAP_y are the ABIS and CAP of AA_y. Based on this final pseudo-distance, the AM can compare the bids given back by the AAs and allocate a newly submitted annotation to the closest agent, following a contract-net protocol.

10.3.3 Conclusion and discussion on the Annotation allocation

The problem of designing a similarity measure or semantic distance comes up in very different branches of artificial intelligence and information retrieval. I shall only recall the closer work.

[Maedche and Staab, 2001] were interested in measuring the extents to which two ontologies overlap and fit with each other at various semiotic levels:

- At the syntactic level: they proposed a syntactic measure based on the edition distance of Levenshtein [Levenshtein, 1966] for weighting the difference between two strings. This distance measures the minimum number of token insertions, deletions, and substitutions required to transform one string into another. In certain cases, this distance may be more robust than the lexicographical one I used (e.g. typing mistakes "azalea" "zaalea"), but it has the same problem of not being semantic at all: the distance between "power" and "tower" is very small, yet their meaning is utterly different.
- At the semantic level: they use the taxonomic structure of two ontologies, calculating the overlap of the semantic cotopy of a concept (sub-types and super-types); the overlap calculation uses the previous edition distance to compare concepts through their labels. However here, I am not interested in this structural distance since I just want to evaluate the additional knowledge that an annotation is bringing to the memory and compare it to the content of the bases.

In the perspectives I shall come back to the problem encountered with this distance, give a number of possible improvements and compare them with other works found in the literature.

10.4 Distributed query-solving

In this section, I present how agents allocate the tasks involved in solving a query. I shall explain how agents use the nested query-ref protocol together with a description of the overlap between the query needs and the statistics over an archive of annotations for distributing sub-tasks in the form of sub-queries. I shall present, first the protocol and the algorithm used for the CoMMA prototype and then a second algorithm partially implemented to validate improvements and support future work.

10.4.1 Allocating tasks

When a user expresses a query, the resolution of this query may involve several annotation bases distributed over several Annotation Archivists. Results may thus be a merging of partial results found by the concerned Annotation Archivists. To determine if and when an Annotation Archivist should participate to the solving of a query, Annotation Archivists calculate the overlap between their ABIS and the properties at play in the query. The result is an OBSIQ (Overlap Between Statistics and Instances in a Query), a light structure which is void if the Annotation Archivist has no reason to participate to the query solving or which otherwise gives the families of properties for which the Annotation Archivist should be consulted. Table 29 gives a short example of an OBSIQ where we can see three families interesting for the resolution process.

Query Properties Overlapping with Statistics	Min	Max
Article → Title → Literal	"Multiple acts ..."	"The incredible..."
Article → Concern → Computer Science		
Person → Name → Literal	"Albertini"	"Hanstucken"
...		

Table 29 OBSIQ (Overlap Between Statistics and Instances in a Query)

The OBSIQ is marshalled using RDF(S)/ XML tags defined in the CORESE namespace and sent by the Annotation Archivist to the requesting Annotation Mediator in charge of co-ordinating the resolution process; an example is given in Figure 85. Using the OBSIQ it requested before starting the solving process, the Annotation Mediator is able to identify at each step of the decomposition algorithm and for each sub-query it generates, which Annotation Archivists are to be consulted.

```
1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <cos:StatOverlap xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:cos="http://www.inria.fr/acacia/corese#">
3
4   <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Title">
5    <cos:RangeType>http://www.inria.fr/acacia/comma#Article</cos:RangeType>
6    <cos:DomainType>http://www.w3.org/2000/01/rdf-schema#Literal</cos:DomainType>
7    <cos:UpperValue>Multiple acts automaton states.</cos:UpperValue>
8    <cos:LowerValue>The incredible conclusion of Pr. Stevenstone</cos:LowerValue>
9   </rdf:Property>
10
11  <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Concern">
12   <cos:RangeType>http://www.inria.fr/acacia/comma#Article</cos:RangeType>
13   <cos:DomainType>http://www.inria.fr/acacia/comma#ComputerScience</cos:DomainType>
14  </rdf:Property>
15
16  <rdf:Property rdf:about="http://www.inria.fr/acacia/comma#Concern">
17   <cos:RangeType>http://www.inria.fr/acacia/comma#Person</cos:RangeType>
18   <cos:DomainType> http://www.w3.org/2000/01/rdf-schema#Literal</cos:DomainType>
19   <cos:UpperValue>Bielin</cos:UpperValue>
20   <cos:LowerValue>Gafreis</cos:LowerValue>
21  </rdf:Property>
22
23  </cos:StatOverlap>
```

Figure 85 OBSIQ marshalled structure

Based on the OBSIQ it received from Annotation Archivists, the Annotation Mediator is able to identify at each step of the decomposition algorithm and for each sub-query it generates, the Archivists to be consulted among the List of Agents Candidates for the Query (LACQ Table 30)

Archivist Agent ID	OBSIQ
Agent ID_1	$OBSIQ_1$
Agent ID_2	$OBSIQ_2$
Agent ID_3	$OBSIQ_3$

...

Table 30 List of Agents Candidates for the Query (LACQ)

10.4.2 Query solving protocol

Agents involved in the allocation of the resolution of a query are:
- the *Interface Controller*, through which the query is built and submitted.
- the *User Profile Manager*, that observes and forward the query.
- the *Annotation Mediator*, that manages the division of tasks, the breaking down and distribution of the query and the merging of partial results to compose the global answer.
- the *Annotation Archivist*, that manages local annotation archives and tries to locally solve sub-queries issued by the Annotation Mediator, using the annotation base it is in charge of

The Interface Controller and User Profile Manager are essentially involved in the emission of the query and I shall not consider them any further here. I also remind that the Annotation Archivist, in this role of query solving, plays a part of the User Profile Archivist role (using the user profiles that may be involved in a query such as documents→written_by→person→name→"gandon") and a part of the Corporate Model Archivist role (using the corporate model that may be involved in a query such as documents→written_by→group→activity→accountancy)

The communication protocol used for the query solving is a composition of the FIPA query-ref protocol to allow multiple stages with sub-queries being exchanged between the Annotation Mediator and the Annotation Archivists. The use of the OBSIQ structure enables the Annotation Mediator to engage in a dialogue only with agents potentially knowledgeable for the current part it is trying to solve.

The protocol is depicted in Figure 86 and corresponds to the following textual description:
1. Through the Interface Controller, the user builds a well-formed and valid query using the ontology O'CoMMA and it submits it.
2. The Interface Controller sends the request for solving the query to the User Profile Manager it is acquainted with.
3. The User Profile Manager contacts an Annotation Mediator. We call it the LocalAM, it is the first one found by the Directory Facilitator, but it could be chosen on more complex criteria such as the workload.
4. The LocalAM calls upon the Directory Facilitator to look for other Annotation Mediators.
5. The LocalAM sends to other Annotation Mediators a query requesting OBSIQs for the given Query.
6. The LocalAM and other Annotation Mediators send a query requesting OBSIQs for the given Query to their Annotation Archivists.
7. The Annotation Archivists send their OBSIQs.
8. The other Annotation Mediators send to the LocalAM the list of non void OBSIQs and corresponding Annotation Archivists (the other Annotation Mediators are matchmakers here).
9. The LocalAM interacts with the Annotation Archivists to solve the query through a sequence of query-ref containing the sub-queries it generates.
10. The LocalAM sends back the global answer to the User Profile Manager.
11. The User Profile Manager answers to the Interface Controller.

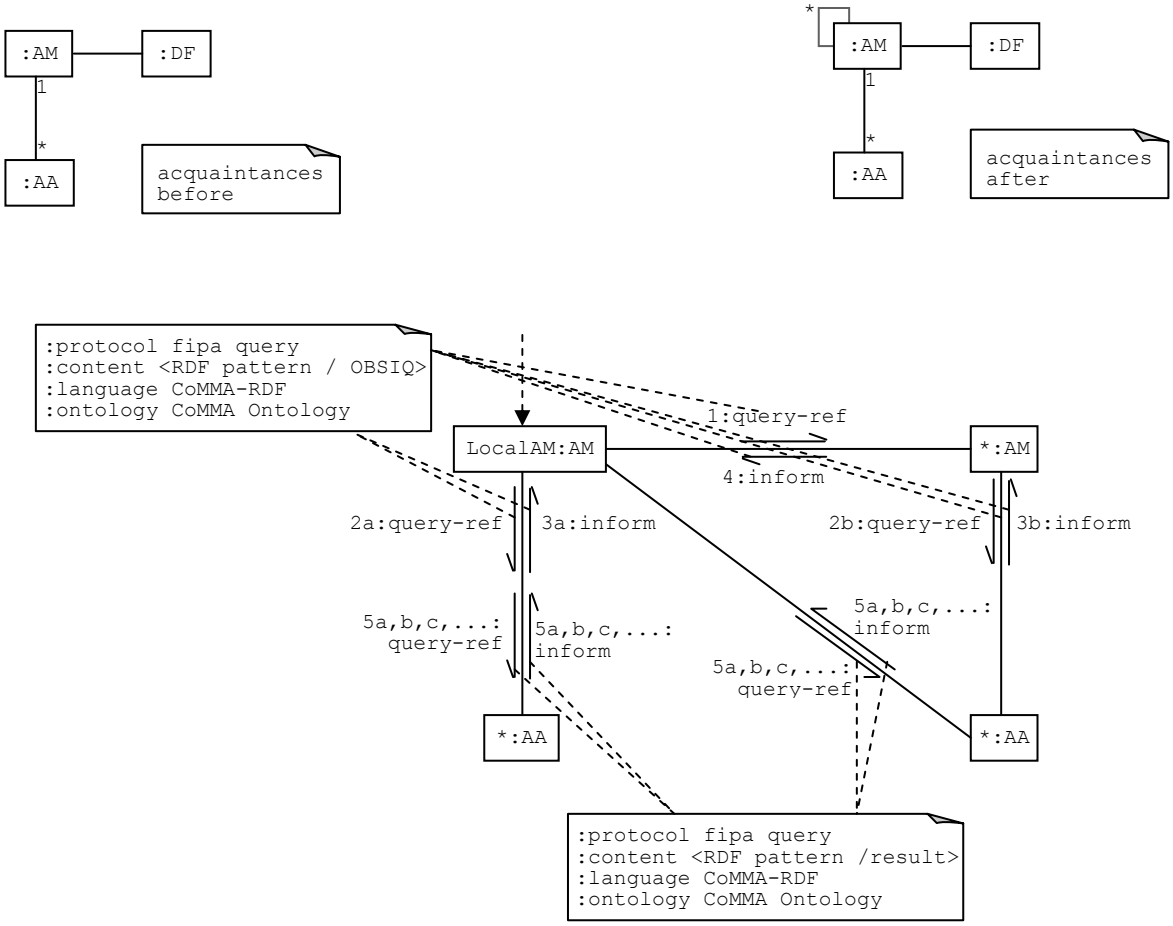


Figure 86 Acquaintance and protocol diagrams in query solving

The whole process relies on the ability to decompose the query and merge the partial results. This process is detailed in the following sections.

10.4.3 Distributed query solving (first simple algorithm)

The decomposition algorithm consists of four stages: the pre-processing for query simplification, the constraint solving, the question answering and the final filtering. These stages, detailed in the following subsections, manipulate the query structure through the Document Object Model (DOM). It is an interface to manipulate an XML document as a tree or, more precisely, as a forest. In our case, the tree structure represents an RDF pattern and contains nodes representing resources or properties, except for the leaves that may be resources or literals (see top part of Figure 88). The resource nodes may have an URI and the Annotation Mediators use them as cut point during query solving to build small sub-queries that can be sent to the Annotation Archivists to gather information that could be scattered in several archives; URI are also joint points to merge partial results.

10.4.3.1 Query simplification

A pre-processing is done on the query before starting the decomposition algorithm. A query may hold co-references. Simple co-references *i.e.,* two occurrences of a variable where one reference is a node of a sub-tree of the query and the other one is the root of another sub-tree, are merged by grafting the second tree on the first one - see Figure 87. Complex co-references would generate distributed constraint problems. They are erased and replaced by simple variables for the duration of the distributed solving; they are then reintroduced at the last stage for the final filtering.

Figure 87 Query simplification process

10.4.3.2 Constraint solving

To cut down the network load, the decomposition starts with the solving of constraints contained in the query, represented by exclamation marks in the top part and the part 1 of Figure 88. The grouping of constraints limits the number of messages being exchanged by constraining the queries as soon as possible.

The Annotation Mediator groups constraints according to the concept instance used for their domain value. It chooses a group of constraints among the deepest ones and creates a sub-query by extracting this sub-tree and asking for the possible URIs of its root concept. Its purpose is to replace this sub-constraint in the global query by the result list of possible URIs for its root, and iteratively reduce the depth of the global query.

Among the candidate Annotation Archivists, the Annotation Mediator identifies the agents concerned by a sub-query thanks to the OBSIQ they provided at the start, and it contacts them to try to solve the sub-query using their local resources:

- If a piece of RDF matches the query and provides the URI of its root concept, the Annotation Archivist sends it to the Annotation Mediator.
- If an annotation violates a constraint, it is dismissed.
- If an annotation answers partially, and if the root concept of the result has a URI, the Annotation Archivist returns the incomplete answer with the URI since the missing part can be found somewhere else thanks to this unique ID.
- If an annotation answers partially, but does not have a URI at the root concept (existential quantification), then the AA does not return it since it cannot be completed elsewhere.
- If an annotation answers the query but does not have a URI for the root concept, the AA returns the whole annotation.

Partial results are merged in the local base of the requesting Annotation Mediator. The Annotation Mediator then reduces the original query using the URIs it has learnt and generates a list of smaller queries. For each one of these queries, it applies this algorithm again until no constraint is left.

10.4.3.3 Question answering

After the constraint solving, the Annotation Mediator has, in its base, complete annotations and partial answers with URIs. Using these unique identifiers, it is able to request the information asked by the user for each one of the resources identified. Therefore, after solving the constraints, the Annotation Mediator emits queries to fill the question fields represented by question marks in the top part and the part 2 of Figure 88. The Annotation Mediator starts from the root of the original query since its potential URIs should have been found by now. It generates a sub-query each time it finds a question during its walk through the tree. Some URIs may still be missing for unconstrained nodes and intermediate queries may have to be issued to solve them.

If the initial query was not constrained at all, there is no URI to constrain the root when starting the question solving. Thus the flow of data to solve it would potentially be too high and could result in a network jam. In that case, the Annotation Mediator switches to a degraded mode and simply asks the Annotation Archivists to solve the whole query locally, potentially loosing some answers.

10.4.3.4 Filtering final results

Once the questions have been answered, the Annotation Mediator projects the original query on the base it has built and extracts the final correct full answers. This stage enables it to finalise the merging of partial results and to take into account the possible cross-references occurring in the constraints, that were discarded during the pre-processing. The Annotation Mediator can then send back the result to the external requester agent.

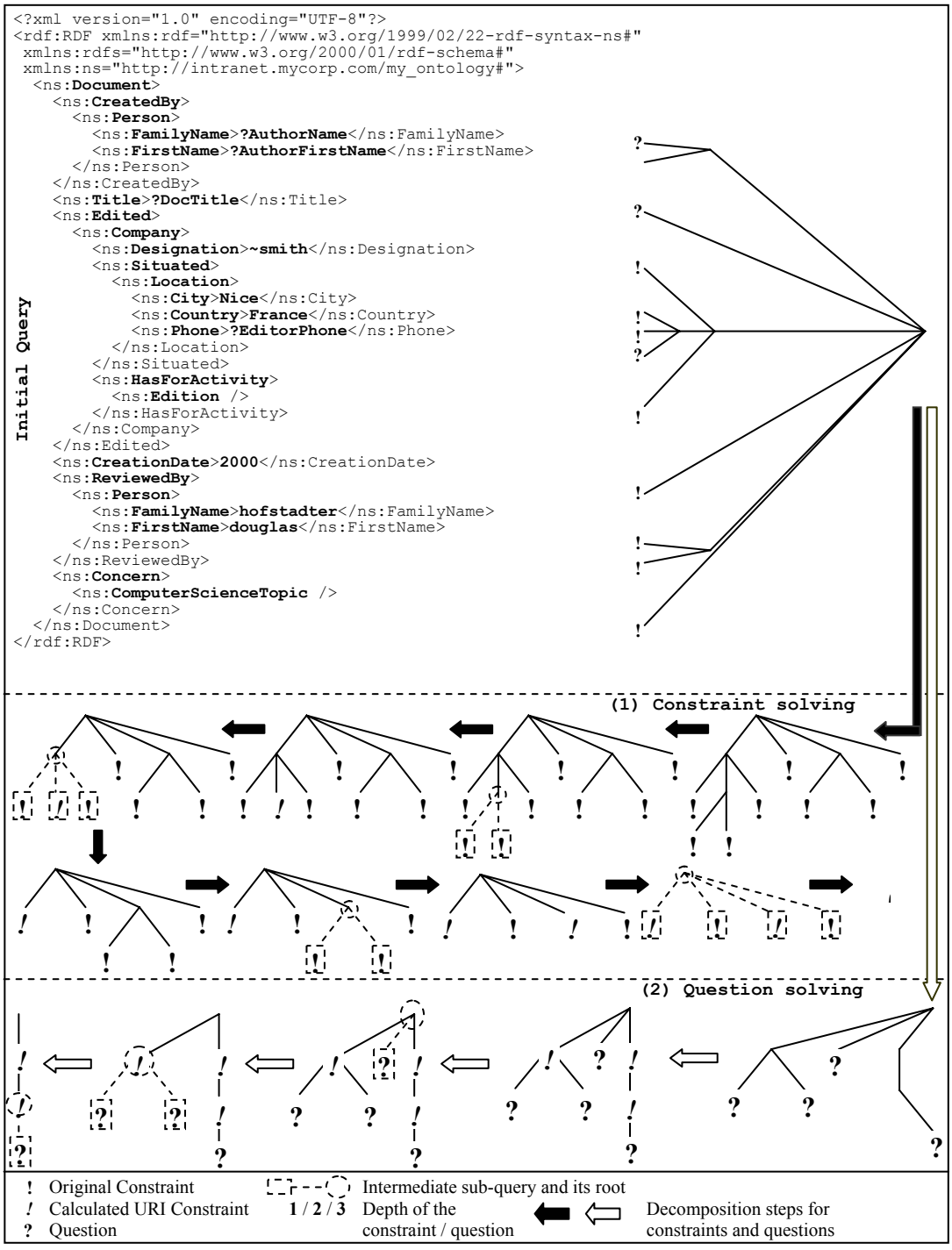


Figure 88 Constraint and question solving

10.4.3.5 Overall algorithm and implementation details

I now give some details of the implementation of this first algorithm. The query decomposition was implemented inside CORESE using two special techniques for walking through the DOM tree:
- complex nested recursive programming was used for propagation of tests and modification in the tree;
- factory-oriented programming was used to generate an up-to-date RDF typing (concept, properties) of the XML nodes of the DOM as the Annotation Mediator walks through the tree and modifies it.

The second point is an interesting way of dynamically casting the RDF nodes. The main classes I used are given in Figure 89; factories are called each time an algorithm moves in the tree and provide instances with the most precise type. Thus polymorphism is used a lot in the design of recursive algorithms to manipulate the query tree.

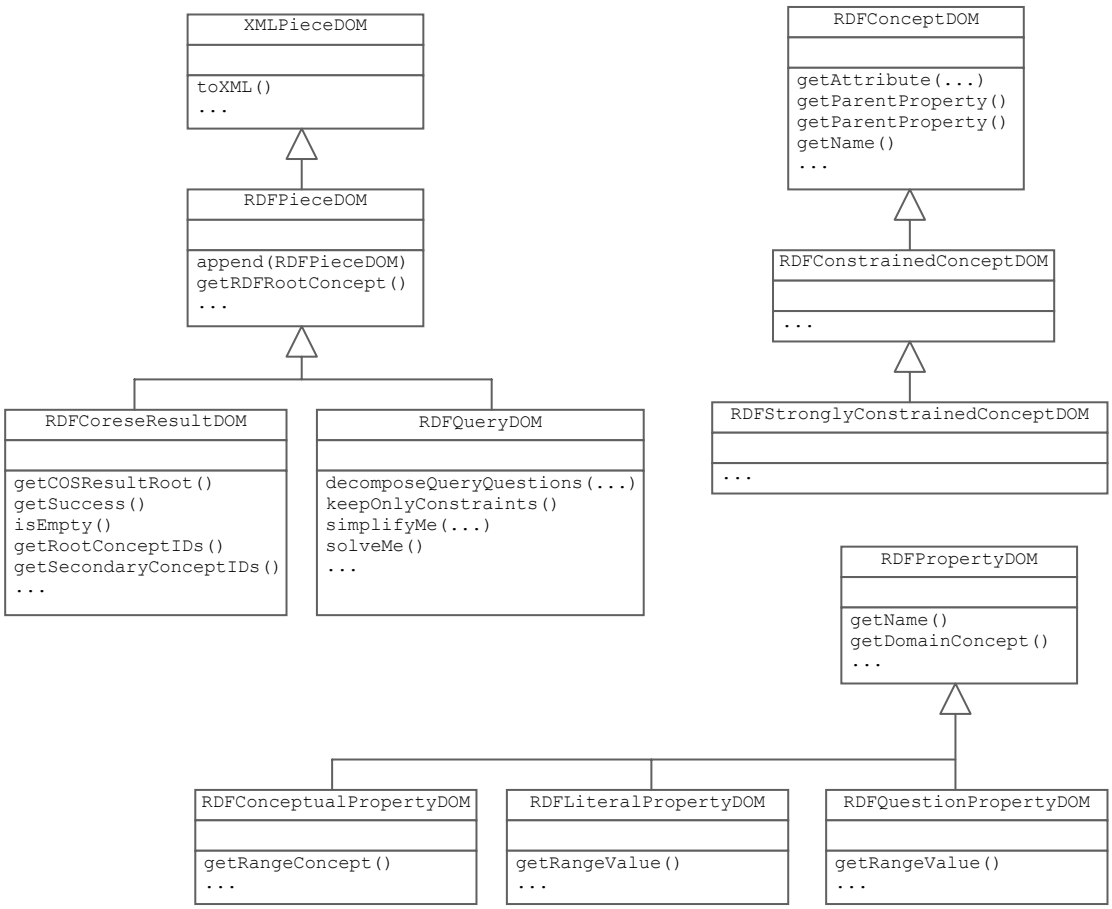


Figure 89 Diagram of main classes for query walking

One of the problems caused by the complex recursive DOM walking and query-solving algorithm is that it could not easily be rewritten in JADE using the nested behaviours since this would have required to 'derecursify' the algorithm. It is a case of agentisation. So instead of designing a query-solving behaviour for the Annotation Mediator, I designed a behaviour able to converse with the query-solving algorithm. I reified the dialogue into a synchronised object and made the Annotation Mediator and its solver run in two different threads dialoguing through a shared synchronised object. It is a case of wrapping a complex algorithm, although this wrapping is close to a transducer approach.

The simplified core of the solver algorithm is given in Figure 90; the lines marked with * make heavy use of sometime nested recursive walking, justifying why "derecursifying" was not an option:

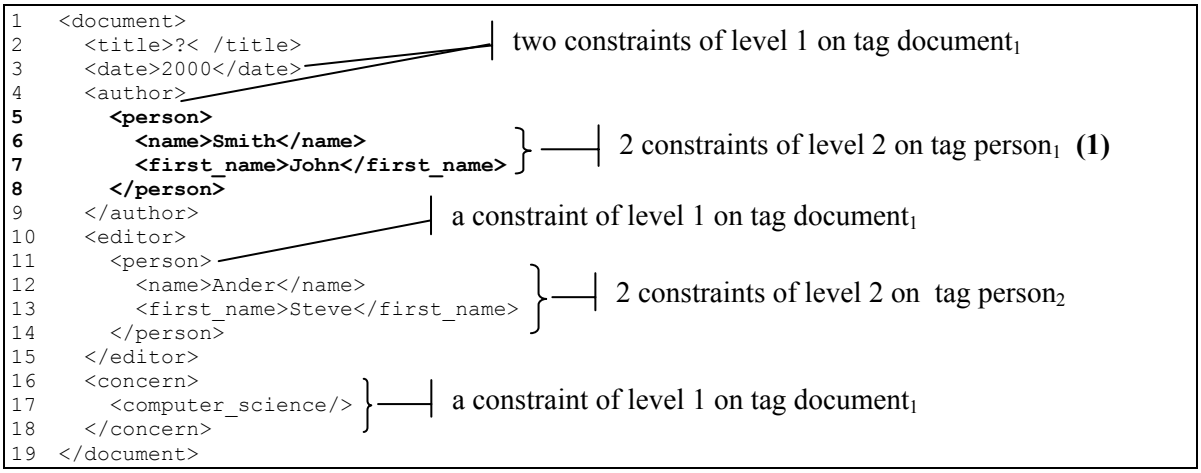
```
1   TempQuery := Simplify(OriginalQuery)
2
3   // Bottom-up constraint-solving
4   RDFConceptDOM ConstrainedConcept := DeepestUnsolvedConstrainedConcept(TempQuery) *
5   FoundURIs := ∅
6   LocalRDFBase  := ∅
7   Results := ∅
8
9   while (ConstrainedConcept ≠ null)
10   {
11    Queries := SubQueriesToGetURIsFor(ConstrainedConcept, FoundURIs) *
12    Results := EmitToConcernedAAs (Queries)
13    LocalRDFBase  := LocalRDFBase ∪ Results *
14    FoundURIs := FoundURIs ∪ {(ConstrainedConcept, ExtractURIs(Results))}
15    Queries := SubQueriesToGetAnnotationsWithExistentialInstancesOf(ConstrainedConcept) *
16    Results := EmitToConcernedAAs (Queries)
17    LocalRDFBase  := LocalRDFBase ∪ Results *
18    MarkAsSolved(ConstrainedConcept)
19    RDFConceptDOM ConstrainedConcept := DeepestUnsolvedConstrainedConcept(TempQuery) *
20   }
21
22  If (FoundURIs=∅)
23   {
24    //SwitchToDegradedMode
25    Results := EmitToAAs(OriginalQuery)
26    return Results
27   }
28  else
29   {
30    // Top-down question-answering
31    CallProcedure QuestionAnswering(GetRootConcept()) *
32    Results := Solve(OriginalQuery, LocalRDFBase)
33    return Results
34   }
35
36  ------------------------------------------------------------------------------------------
37
38  Procedure QuestionAnswering(RDFConceptDOM Concept)
39   {
40    For each Q ∈ QuestionsAttachedTo(Concept)
41     {
42      Queries := SubQueriesToGetAnswer(Q, FoundURIs) *
43      Results := EmitToConcernedAA (Queries)
44      LocalRDFBase  := LocalRDFBase ∪ Results *
45     }
46    For each C ∈ UnsolvedConceptsInPropertiesOf(Concept)
47     {
48      Queries := SubQueriesToGetURIsFor(Q, FoundURIs) *
49      Results := EmitToConcernedAA (Queries)
50      LocalRDFBase  := LocalRDFBase ∪ Results *
51      FoundURIs := FoundURIs ∪ {(ConstrainedConcept, ExtractURIs(Results))}
52     }
53    For each C ∈ ConceptsInPropertiesOf(Concept)
54      QuestionAnswering(C) *
55   }
```

Figure 90 Query solver simplified core algorithm

10.4.4 Example and discussion

The following section gives an example of the process performed by the query breaking down algorithm.

To restrain data flow, the algorithm of the Annotation Mediator starts with the constraints. First it groups constraints according to their depth and their 'domain tag':




```
1   <document>
2     <title>?< /title>
3     <date>2000</date>
4     <author>
5       <person>
6         <name>Smith</name>
7         <first_name>John</first_name>
8       </person>
9     </author>
10    <editor>
11      <person>
12        <name>Ander</name>
13        <first_name>Steve</first_name>
14      </person>
15    </editor>
16    <concern>
17      <computer_science/>
18    </concern>
19  </document>
```

Figure 91 Anatomy of a sample query

Secondly, it chooses a group of constraints among the deepest ones and having the same 'domain tag'. It creates a sub-query based on this group. For instance in Figure 91, it chooses group (1) and generates the sub-query of Figure 92:

```
1   <person ID="?">
2     <name>Smith</name>
3     <first_name>John</first_name>
4   </person>
```

Figure 92 Sample sub-query

The obtained sub-query is submitted to the Annotation Archivists the OBSIQ of which has a non null overlap with it. The reason for grouping constraints is to avoid too much or too big messages to be exchanged by trying to constrain the request as soon as possible.

Chosen Annotation Archivists try to solve the query using their base:

- If a piece of annotation corresponds perfectly and the top concept has a URI, the Annotation Archivist sends this piece of RDF to the Annotation Mediator (Figure 93).

```
1   ...
2     <title>Effectiveness of Factories in AOP</title>
3     <author>
4       <person rdf:about="http://www.xyztech.com/#jsmith">
5         <name>Smith</name>
6         <first_name>John</first_name>
7       </person>
8     </author>
9   ...
```

```
1   <person ID="http://www.xyztech.com/#jsmith">
2     <name>Smith</name>
3     <first_name>John</first_name>
4   </person>
```

Figure 93 Perfect match of sub-query with root URI available

- If an annotation explicitly violates a constraint the annotation is dismissed as shown in Figure 94.

```
...
  <title>Effectiveness of Factories in AOP</title>
  <author>
    <person rdf:about="http://www.xyztech.com/#jsmith">
      <name>Smith</name>
      <first_name>Stephen</first_name>
    </person>
  </author>
...
```

Figure 94 Constraint violated in sub-queries

- If an annotation answers partially, if it does not violate a constraint and if it has a URI, then the Annotation Archivist returns the incomplete answer (Figure 95) with the URI since it can be completed elsewhere through this reference.

```
...
  <title>Effectiveness of Factories in AOP</title>
  <author>
    <person rdf:about="http://www.xyztech.com/#jsmith">
      <name>Smith</name>
    </person>
  </author>
...
```

```
<person ID="http://www.xyztech.com/#jsmith">
  <name>Smith</name>
</person>
```

Figure 95 Partial match in a sub-query with URI

- If an annotation answers partially, if it violates no constraint, but has no URI, then the Annotation Archivist does not return it since it cannot be completed elsewhere as shown in Figure 96.

```
...
  <title>Effectiveness of Factories in AOP</title>
  <author>
    <person>
      <name>Smith</name>
    </person>
  </author>
...
```

Figure 96 Partial match in a sub-query without URI

We can see here that the notion of *key* as it is used in data bases should be studied and improved.

- If an annotation perfectly answers the query, but has no URI for the "domain tag", the Annotation Archivist returns the whole annotation as shown in Figure 97.

```
<report rdf:about="http://www.xyztech.com/report.html">
  <title>Effectiveness of Factories in AOP</title>
  <author>
    <person>
      <name>Smith</name>
      <first_name>John</first_name>
    </person>
  </author>
</report>
```

Figure 97 Partial match in a sub-query without URI

A possible improvement would be to use the depth of the constraint in order not to bring back too deep annotations. In this case, the depth being 2, we would cut anything deeper than 2. Another approach would be to cut at the first URI encountered thus sending back a perfectly autonomous context around the piece of annotation interesting for the solving.

These improvements have not been implemented, but we shall see, in the next section, another algorithm that does not suffer from that problem.

Partial results are merged in the base of the Annotation Mediator using URI as joint point. Only complete answers will be used in the rest of the process, *i.e.*:
- Exact answers with a URI as in:

```
...
<person rdf:about="http://www.xyztech.com/#jsmith">
  <name>Smith</name>
  <first_name>John</first_name>
</person>

<person rdf:about="http://www.abc-consult.com/#john.smith">
  <name>Smith</name>
  <first_name>John</first_name>
</person>
...
```

Figure 98 Base of exact answers with the exact URI

- Complete blocks of annotations possibly without URI:

```
...
<document>
  <author>
  <person>
    <name>Smith</name>
    <first_name>John</first_name>
  </person>
  </author>
  <summary> Once upon a time in a start-up, there was a nice... </summary>
</document>

<report rdf:about="http://www.xyztech.com/report.html">
  <author>
    <person>
      <name>Smith</name>
      <first_name>John</first_name>
    </person>
  </author>
  <title>Effectiveness of Factories in AOP</title>
</report>
...
```

Figure 99 Base of complete answers without the exact URI

From these firsts results, the Annotation Mediator reduces the original query: using what it has learned, it can generate new smaller queries as the one given in Figure 100:

```
1   <document>
2     <title>?< /title>                          ┐ two constraints of level 1 on tag document1
3     <date>2000</date>                          ┘
4     <author>
5       <person rdf:about="http://www.xyztech.com/#jsmith"/>
6     </author>
7     <editor>                                   ┐ a constraint of level 1 on tag document1
8       <person>                                 ┘
9         <name>Ander</name>                     ┐
10        <first_name>Steve</first_name>         ┘ 2 constraints of level 2 on tag person2 (2)
11      </person>
12    </editor>
13    <concern>                                  ┐
14      <computer_science/>                      ┘ a constraint of level 1 on tag document1
15    </concern>
16  </document>
```

Figure 100 Example of a reduced query

The Annotation Mediator starts again from this new set of queries, back to the previous sub-query solving process; for instance in Figure 100 it will choose the group of constraints **(2)** and reduce the query again. The last step of this iterative process is when only level 1 constraints are left, an example is shown in Figure 101:

```
1   <document>
2     <date>2000</date>
3     <author>
4       <person ID="http://www.abc-consult.com/#john.smith" />
5     </author>
6     <editor>
7       <person ID="http://www.uvw-edition.com/#ander" />
8     </editor>
9     <concern><computer_science/></concern>
10  </document>
```

Figure 101 Example of last stage in query reduction

From these last queries the Annotation Mediator submits the last set of sub-queries such as the one in Figure 102:

```
1   <document ID="?">
2     <date>2000</date>
3     <author>
4       <person ID="http://www.abc-consult.com/#john.smith" />
5     </author>
6     <editor>
7       <person ID="http://www.uvw-edition.com/#ander" />
8     </editor>
9     <concern><computer_science/></concern>
10  </document>
11  ...
```

Figure 102 Example of last stage sub-query

Once this is finished, the base of the Annotation Mediator contains all the needed URI to start completing the question parts of the original query.

After solving the constraints, the Annotation Mediator uses the known URI to emit queries to fill the fields to be given in the answers (question marks).

For instance, from the URI in `<document ID="http://www.efg-books/report.html" />`, the Annotation Mediator can send to the Annotation Archivists the query of Figure 103.

```
1   <document ID="http://www.efg-books/report.html">
2     <title>?</title>
3   </document>
```

Figure 103 Example of sub-query for the question answering

However some URIs may still be missing: for instance in the query in Figure 104, there is no constraint on the author, therefore, the Annotation Mediator does not have any URI for the person.

```
1   <document>
2     <title>?</title>
3     <date>2000</date>
4     <author>
5       <person>
6         <name>?</name>
7         <first_name>?</first_name>
8       </person>
9     </author>
10    <editor>
11      <person>
12        <name>Ander</name>
13        <first_name>Steve</first_name>
14      </person>
15    </editor>
16    <concern>
17      <computer_science/>
18    </concern>
19  </document>
```

Figure 104 Query with an unconstrained concept

The Annotation Mediator must therefore, solve the questions starting from the known URIs, submitting queries such as the one in Figure 105.

```
1   <document ID="http://www.ukbooks/review.html">
2     <author>
3       <person ID="?" />
4     </author>
5   </document>
```

Figure 105 Query to solve URI of unconstrained concept during question-answering

Then the mediator can proceed with the solved URIs as in the example of Figure 106.

```
1   <person ID="http://www.xyztech.com/#thomas_sanderstone">
2     <name>?</name>
3     <first_name>?</first_name>
4   </person>
```

Figure 106 Following query to answer the questions

Once the questions have been answered, the local base of the Annotation Mediator contains all the information to solve the original query.

10.4.5 Distributed query solving (second algorithm)

10.4.5.1 Discussion on the first algorithm

First algorithm made some naive choices and left some open issues:

- The worst case that could happen is the one of a query with no constraint as in Figure 107.

```
1   <document>
2     <title>?</title>
3     <author>
4       <person>
5         <name>?</name>
6         <first_name>?</first_name>
7       </person>
8   </document>
```

Figure 107 Unconstrained query

If such a query was decomposed into unconstrained sub-queries, the network will be overloaded since they may generate huge answers (e.g. all the persons with their names). The problem is that there is no simple way to avoid this effect since the query is by itself too large. Other projects have used 'watchdogs' agents (see section 4.4.1) to allow such queries while monitoring high traffic queries and prevent them from jamming the network.

- In the case of a query containing only isolated concepts such as in Figure 108, the algorithm does not work properly; yet these queries are perfectly acceptable for CORESE.

```
1   <document/>
2   <KnowledgeEngineering/>
```

Figure 108 Vertex-based query

Such a query should be recognised and solved in degraded mode. It was not done in CoMMA since the query interface of the Interface Controller does not allow the user to create such a query.

- In the case of a query with cross constraints as in Figure 109, decomposition of constraints would be too complex to be distributed.

```
1   <document>
2     <title>~?x</title>
3     <author>
4       <person>
5         <name>?x</name>
6         <first_name>fabien</first_name>
7       </person>
8     <date>1999</date>
9   </document>
```

Figure 109 Query with cross references

As explained in the previous algorithm, in the first prototype the Annotation Mediator uses different variables for decomposition and when it finally projects the original queries it uses the cross constraints variables to filter the results. However the simplification of the query makes it much less constrained and augments the number of messages exchanged for the resolution. Improvements could be found in considering the work done, for instance, on the semi-joint operator used in Distributed Data Bases [Özsu and Valduriez, 1999].

10.4.5.2 Improved algorithm

To start improving the algorithm, I considered one of the sources of multiplication of messages: the existential quantificator. Indeed one of the root problem of my first algorithm is that it really treated the existential quantificator encountered in an annotation as a universal statement of the existence of an instance. For example in Figure 110, there are two existential quantifications: one says that "*there exists a report* that has for title Effectiveness of..." and another one says that " *there exists a person* called John Smith who wrote ...". The lack of URIs reduces the instances to the mere statement of the existence of an instance.

```
1   <report>
2     <title>Effectiveness of Factories in AOP</title>
3     <author>
4       <person>
5         <name>Smith</name>
6         <first_name>John</first_name>
7       </person>
8     </author>
9   </report>
```

Figure 110 Existential quantification in an annotation

However these statements have a context of annotation and even more, they are issued by an Archivist Agent uniquely identified by its address. The idea is to recreate URIs for these statements, denoting the source of the existential quantification; these URIs being based on the address of the agent and the internal generic IDs of CORESE. The annotation in Figure 110 would thus virtually be considered as the annotation in Figure 111:

```
1   <report ID="acacia_archivist@fapollo:1099/JADE#genID54">
2     <title>Effectiveness of Factories in AOP</title>
3     <author>
4       <person ID="acacia_archivist@fapollo:1099/JADE#genID79">
5         <name>Smith</name>
6         <first_name>John</first_name>
7       </person>
8     </author>
9   </report>
```

Figure 111 URIs for existential statements

Using this approach, the Annotation Mediator knows which agent issued a statement and thus if further sub-queries use this ID, they will be sent only to the relevant agent. There is no longer a need for retrieving full annotations when no URI is available. Moreover, the previous algorithm was issuing two types of queries (one to get answers with URIs and one to get answers without URIs): this is no longer needed.

The second point addressed by the improved algorithm is the choice of the order in solving the constraint. The previous algorithm chose the first deepest constrained concept to start with; this is because most constraints are given in the leaves by literal values or imposed regular expressions. The new algorithm tries to determine the most constrained concept in the query and starts with this concept in order to cut down the number of results as soon as possible.

The third point is that instead of generating multiple sub-queries to cover all the possible URIs for a concept, the solver now generates one sub-query with a disjunctive list of URIs thus saving a large number of messages.

Together these three improvements now allow me to reduce the number of sub-queries, the size of results and to detect sooner if the conjunction of the constraints in a query cannot be satisfied: this corresponds to a case where no URIs at all (original or generated) could be found for a constrained concept.

This algorithm also supports future improvements such as the use of heuristics to sort the types of constraints and to start with the strongest ones: "known URI" > "imposed literal values" > "literal value constrained by regular expression" > ...

The new algorithm has been partially implemented and tested (a centralised version). For information, a simplified version of the core algorithm of the solver is given in Figure 112:

```
1   Let KnownURIs := ∅ a set of couples (Concept,{URI}) to memorise found URIs for a query
2   Results := ∅
3
4   ConstrainedConcept := GetStrongestConcept(OriginalQuery)
5   if (ConstrainedConcept = null)
6    {
7     //SwitchToDegradedMode
8     Results := EmitToAAs(OriginalQuery)
9     return Results
10   }
11  else
12    while(ConstrainedConcept ≠ null)
13     {
14      TempURIs := null // URIs found for the ConstrainedConcept
15      // Deal with conceptual properties
16      For each C ∈ {Concept linked to ConstrainedConcept by a property P}
17       {
18         if ∃ (C,URIs) ∈ KnownURIs
19         {
20          Query := SubQueryToGetURIs(ConstrainedConcept,TempURIs,P,C,URIs)
21          Results := EmitToConcernedAA (Query)
22          LocalRDFBase  := LocalRDFBase ∪ Results
23          if (TempURIs = null)
24            TempURIs := ExtractURIs(Results)
25          else
26            TempURIs := TempURIs ∩ ExtractURIs(Results)
27          if (TempURIs=∅)
28            return    // there is no answer to the query
29         }
30      // Deal with literal properties
31      For each L ∈ {Literal constraint linked to ConstrainedConcept by a property P}
32       {
33        Query := SubQueryToGetURIs(ConstrainedConcept,TempURIs,P,L)
34        Results := EmitToConcernedAA (Query)
35        LocalRDFBase  := LocalRDFBase ∪ Results
36        if (TempURIs = null)
37          TempURIs := ExtractURIs(Results)
38        else
39          TempURIs := TempURIs ∩ ExtractURIs(Results)
40        if (TempURIs=∅)
41          return    // there is no answer to the query
42       }
43      // Deal with questions
44      For each Q ∈ {Literal question linked to ConstrainedConcept by a property P}
45       {
46        Query := SubQueryToGetAnswer(ConstrainedConcept,TempURIs,P,Q)
47        Results := EmitToConcernedAA (Query)
48        if (Results=∅ and ¬ OptionalQuestion(Q))
49          return; // A non optional question could not be answered
50        LocalRDFBase  := LocalRDFBase ∪ Results
51       }
52      MarkAsSolved(ConstrainedConcept)
53      KnownURIs := KnownURIs ∪ {(ConstrainedConcept, TempURIs)}
54      ConstrainedConcept := GetStrongestConcept(OriginalQuery)
55     }
56
```

Figure 112 New query solver simplified core algorithm

The algorithm works by propagation: as soon as a concept is solved (*i.e.,* its potential URIs are known), its neighbours become constrained by their relation with a concept having known URIs, and so on. If no constrained concept is found, a pre-processing searches for loosely constrained concepts to initiate the process; this is where the heuristics could be introduced to sort constraints.

10.5 Annotation push mechanism

You affect the world by what you browse.
— Tim Berners Lee

Finally, one last mechanism is triggered by annotation management agents: the push of newly submitted annotations to concerned users. This functionality is triggered by the annotation mediator.

In the actual prototype, a simplified version of the complete scenario was implemented by our partners of ATOS-Origin:

- *User profile*: Users are able to modify their user profile through the Interface Controller in order to make explicit their interests (personal interest, work interest, etc.). This operation can be done at any time since the user may change projects or departments at any time. The result is an annotation about the user, as shown previously, stored by the User Profile Manager. These interests constitute implicit Communities of Interest (CoIn) and the statement of an interest can be seen as a registration of the user to a CoIn.
- *User interest filters*: Simple queries are derived from the user profile description, exploiting the properties like "hasForWorkInterest", "hasForPersonalInterest". The different topics which are pointed out by these properties are used to build queries used as filters on the newly added annotations. The automatically generated queries test the concern field of the annotation: thus if, for instance an employee has for work interest the topic "Java programming", then the query will request for documents that concern "Java programming".
- *Proactive matching*: these queries are used as filters on new annotations to decide for each profile if the new annotation is of interest and should be stored as a suggestion in the profile or if it should be discarded.
- *Learning and presentation*: suggestions are sorted by the machine learning algorithm and presented at login to the user. Feedback from the user is used to improve the learned preferences.
- In the future version of CoMMA, a more complex way to generate queries can be envisaged. This could rely on inference mechanisms and concern any kind of annotation (people, enterprise model changes,…).

The different agents that are mainly involved into the push mode are the following:

- *The Annotation Mediator* (AM), in charge of offering a notification service for other agents interested in new annotation events.
- *The User Profile Manager* (UPM), in charge of registering for new annotation events.
- *The User Profile Archivist* (UPA). in charge of providing filtering service based on profiles.

The textual description of the interactions depicted in Figure 113 is the following:

- The User Profile Manager registers with the Annotation Mediator to get notified of any new annotation.
- Each time a new annotation is notified, the User Profile Manager requests the User Profile Archivist to filter the annotation with the derived queries and to add suggestions.
- Each time a user logs in, the Interface Controller requests the User Profile Manager to retrieve the profile and the push suggestions are displayed.
- Additional interactions take place for the feedback and machine learning aspects, that are not depicted in the diagrams.

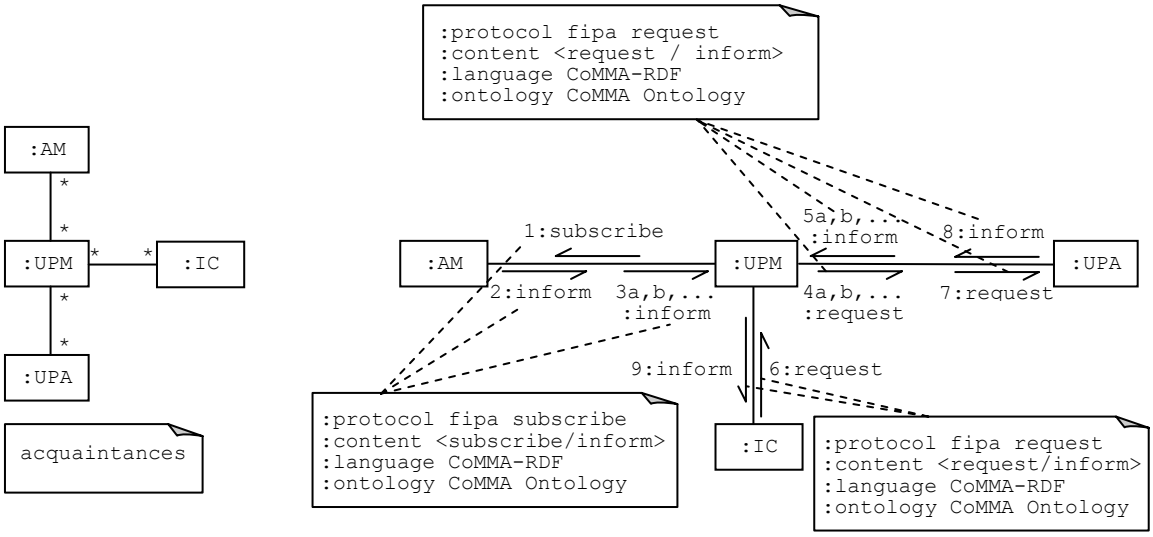


Figure 113 Push mode interactions

The User Profile Archivist uses the CORESE engine to apply the filtering queries generated from the profiles to filter the annotations and memorise the one matching a query in the suggestion list of the profile.

10.6 Conclusion and discussion

The stake of the work presented in this chapter was to make a proof of concept of the interest of merging ontology and distributed artificial intelligence to provide distributed mechanisms managing distributed knowledge. Knowledge distribution is both the spatial property of knowledge of being scattered about and the act of diffusing knowledge. I focused on the tasks of managing both facets of distribution to allocate storage and distribute exploitation and since they are linked to one another in their performances, I chose two complementary strategies to implement them:

- the *allocation of an annotation* is based on a contract-net where bids are the result of a distance between the semantic contribution of the annotation and the description of the semantic contribution of an archive to the memory content. This protocol tends to maintain the specialisation of the base.
- the *solving of a query* exploits the description of the semantic contribution of an archive to the memory to allocate sub-tasks to the relevant agents in charge of the archives. This protocol tries to limit the number of messages exchanged during the distributed query solving process while enabling the answers to be found even if the needed information is split over several bases.

The implementation was done and tested in JADE; the debugger tools of JADE provide a message sequences diagram. Figure 114 and Figure 115 show and comment the snapshots of two trial sequences respectively for the allocation of an annotation and the solving of a query. Both sequences involve, one Interface Controller (IC: localIC), one User Profile Manager (UPM: localUPM), one Annotation Mediator (AM), three Annotation Archivists, one of which is part of a User Profile Archivist role (AA: localAA1, localAA2 and UPA: profileAA).

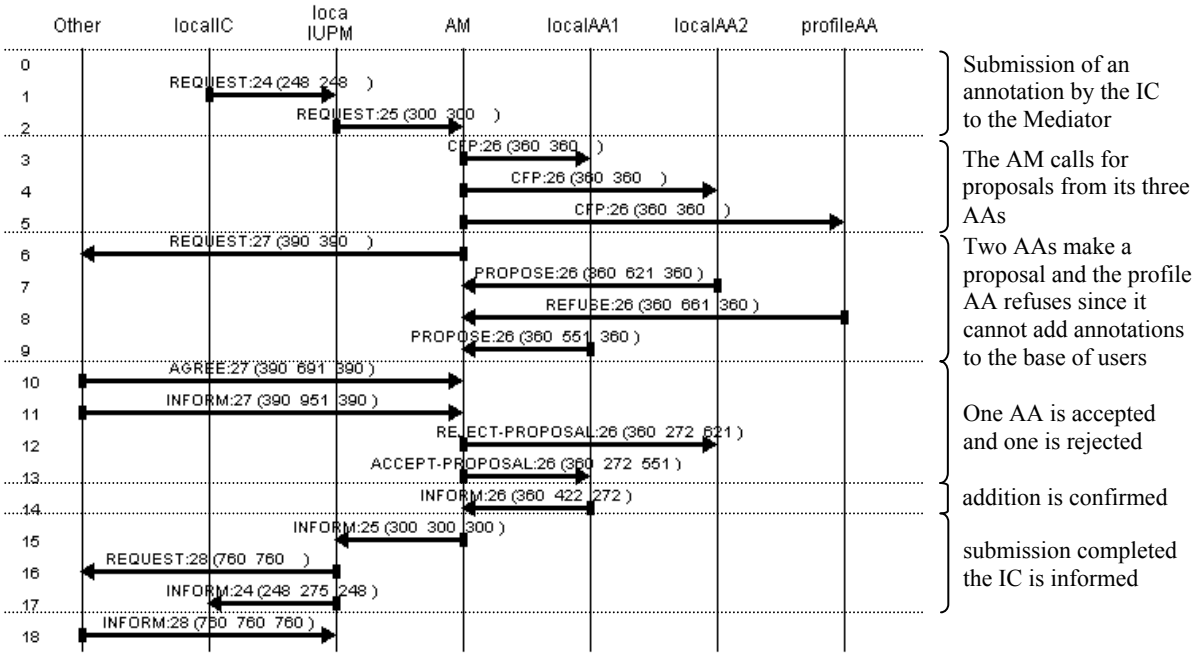


Figure 114 Snapshot of an annotation allocation sequence diagram

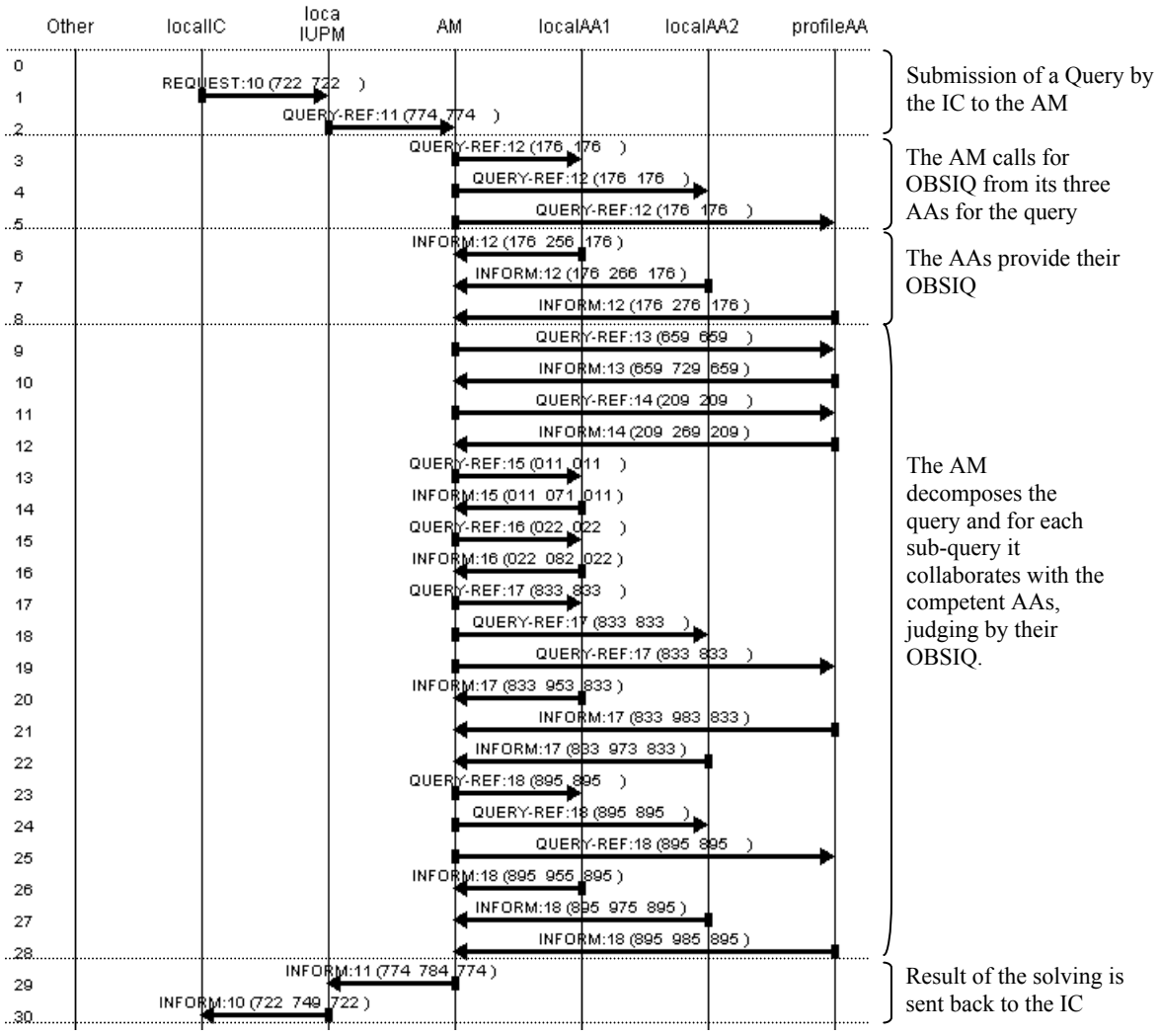


Figure 115 Snapshot of a query solving sequence diagram

In the first case, the ontology is used as a shared space enabling to define a shared distance function the results of which can be compared in a distributed protocol. In the second case, the ontology provides primitives to describe the knowledge possessed by an agent and thus to rule on the pertinence of its involvement in a co-operation. Thus, it clear that here, the ontology is a corner stone of the intelligent distributed mechanisms to manage distributed knowledge.

The ontological consensus lays a semantic foundation on which further inferential consensus may be built. In the distributed artificial intelligence paradigm, the ontology can provide the keystone for designing social co-operation mechanisms.

Part - IV

Lessons learned & Perspectives

Failure is the only opportunity
to begin again more intelligently.
— Henry Ford (1863 - 1947)

11 Evaluation and return on experience

If ease of use was the only valid criterion, people would stick to tricycles and never try bicycles
— D. Engelbart

Because research is a never ending story, what I presented here is only a snapshot of my work at the time I wrote this document. This snapshot has its flaws and I shall present in this chapter some critics and evaluation elements from what I have learnt from this work and try to objectively evaluate this contribution. I shall first give the implementation status of CoMMA and what was left undone. The second part details the official evaluation of CoMMA, that concerns the work presented here. It will first explain the trial process to get an evaluation from the end-users. Then, it will detail the criteria used by the members of the consortium to evaluate the work done and I shall apply it to the three aspects that were essentially developed in this Ph.D.: the ontology aspect, the semantic Web aspect and the multi-agent system aspect.

11.1 Implemented functionalities

Before evaluating what was achieved, I recall what was implemented in the last prototype of CoMMA.

A session starts with the authentication of the user implemented with a classical login and password process. If successful, the login proceeds with the loading of the ontology, the corporate model and the user profile by the activated Interface Controller Agent. The loading of the user profile triggers the display of pushed information that was added to the profiles.

The user interface allows the users to consult pushed documents the list of which is displayed in the start-up window. They can consult the corresponding resources and give their feedback on the suggestions.

The users can browse and modify their profiles as an annotation about themselves. For this reason, the same GUI environment as for annotation is used to visualise, modify, and save a user profile.

The users can browse the ontology using the set of stylesheets that were developed for ontology engineering.

The users can create new annotations or formulate a query. To do so, interfaces organised according to a "Z route" enable the users to instantiate concepts and properties defined in the ontology. The only difference is that the annotation is about an initially provided URI, whereas the query contains variables to express the annotation pattern to be found. The "Z route" GUI guides the user in moving from terms to concepts and relations, from concepts and relations to structured facts and then to populated attributes. Ready-to-use annotation and query templates are available and saved in the user profile. Users start with a predefined set of favourite annotations depending on their profile and they have the possibility to add new favourite ones.

Behind the front-end, every exchange and feedback of the user are analysed by a machine learning algorithm embedded in the User Profile Manager agent. This is used to improve the relevancy of the results, taking into account the user's areas of interest. The system learns to sort the results before presenting them to the users.

To archive the created annotations in one of the distributed annotation bases, I implemented a specific algorithm in charge of selecting the most relevant bases where the new annotations will be saved. To trigger the push mode, each time a new annotation is added, a notification is issued and a process starts up to compare the new annotation with all the user profiles to check if some users can be interested in the new information resource. If a user profile matches the annotation, the suggestion is added to the profile so that it will be pushed at next login.

To be solved, a submitted query is matched with annotations present in different annotation bases through the CORESE semantic search engine. A distributed algorithm has been implemented to take into account the fact that annotations are distributed in different annotation archives and make the agent co-operate over the solving process.

An ontology-dedicated agent type stores the ontology and makes it available to the whole system in order to support the other roles. Likewise, agent types dedicated to the corporate model and the user profiles store the annotations about the organisational entities and the persons, and make them available to other agents.

11.2 Non-implemented functionalities

The current version of the CoMMA system has been limited to a set of core functionalities that enabled us to test the support of such a system to the considered corporate memory management scenarios, taking into account the short time allocated to the development of the prototype and its trials.

While they were discussed during the specification of design, some functions have been disregarded for the sake of feasibility of the project:

- *Corporate model edition*: the corporate model browser has not been implemented for lack of time and therefore, corporate models have been generated directly in RDF for the trials.
- *User profile creation*: the creation of a new user profile has not been implemented in the current version.
- *Security and administration*: management of security and administrator rights and creation of new user profiles by privileged users.
- *Web browser integration*: Full integration of the Interface Controller agent and the users' Web Browser for a seamless integration to an intraweb portal (plug-in development had been envisaged, but was too much time-consuming).
- *Analysis of queries for the push mode*: collect failed queries to periodically rerun them or alert experts of an identified area that can provide the user with more precise information; the set of targeted experts could be found by matching the query content to the user profiles.
- *Improvement of the push mode*: just one type of implicit query is dealt with by the current version. This query enables to match people interested in some topics, to documents that concern these topics. Other queries of this kind can be proposed (relating document to people's activities, etc.). Moreover, the addition of a new user profile should trigger the push mode as well. This improvement would enable to make a newcomer access all interesting documents already present in the corporate memory. In order not to saturate the newcomer at the first login, a scheduled process could be envisaged.
- *Proxy function of Annotation Mediator*: the core of the distribution protocols for allocation and querying annotations have been implemented, but not the nested aspect *i.e.,* Annotation Mediators contacting other Annotation Mediators. This function would only require some development time and was not judge necessary for trials.

Other extensions will be envisaged further in this section, but they were never part of the CoMMA workplan.

11.3 Official evaluation of CoMMA

The following sections are based on the evaluation sessions and reports of CoMMA. I shall focus on the points of interest for us here; there were many more aspects evaluated in CoMMA.

11.3.1 Overall System and Approach evaluation

The prototype was evaluated by end-users from a telecom company (T-Nova Deutsch Telekom) and a construction research centre (CSTB) through two trials (at the eighth month and the twenty-second month) during the two-year project. As the design specifications, the evaluations performed were influenced by scenario analysis [Caroll, 1997], the two generic scenarios being "technology monitoring", and "new employee integration".

The very last prototype was also presented and discussed during an open day at the end of the twenty- third month.

Since the MAS architecture is used to integrate a lot of different components that are vital to the system (GUI, semantic search engine, XLST engine, machine learning algorithm, etc.) if one of them goes wrong, the whole system evaluation may be hampered. Indeed, it is extremely hard to evaluate a component independently from the other components on which it may rely.

The multi-agent paradigm being used as an integration technology, the evaluation of the individual components integrated may prove to be difficult if only the final system is considered in evaluation.

A consequence is, for instance, that if there is a problem at the interface level, it may hamper a good evaluation of the information retrieval capabilities as a whole. Thus three types of evaluations have been distinguished:
- an evaluation of the technical solution built for the CoMMA system;
- an evaluation of the methodology employed to design the CoMMA system;
- an evaluation of the CoMMA system from the user's point of view.

This first part concentrates on the overall approach as it was perceived by users, *i.e.,* the third type of evaluation distinguished above. It gives a short account of the evaluation work supervised by Alain Giboin. The two other types will be addressed in the following sections concerning specific aspects of the system [Giboin *et al.*, 2002].

User evaluation was two-fold: the *usefulness* and the *usability*:
- The *usefulness* evaluation was performed with regard to the functionalities of CoMMA (functional evaluation).
- The *usability* evaluation was performed with regard to the CoMMA interfaces or GUIs (interface evaluation). We shall here discuss user evaluation, which was the most critical.

11.3.1.1 First trial evaluation

In the first trial, four T-Nova employees participated in the functional evaluation based on the New Employee Integration scenario. Two of these employees were real newcomers. The two other ones were asked to put themselves in newcomers' shoes, and to remember the time they were T-Nova newcomers. Both types of users belonged to the same research team. Three CSTB employees participated in the functional evaluation based on the technology monitoring scenario. They were archivists specialised in technological and documentary monitoring.

From the end-user viewpoint, CoMMA was judged as a really useful, if not usable, system. It was deemed useful to the organisation (groups), and to its members (individuals). It meets both group and individual needs. Some suggested adaptations and extensions to the existing functionalities were suggested to provide a complete environment for newcomers or technology watchers.

Usability problems rested on the following issues that are symptomatic of communication breakdowns between designers and end-users [Giboin *et al.*, 2002]:

- A first issue was the *relevance of the contents of the scenarios*. Because one of our "end-user partner" (specialised in telecommunication industry) withdrew from the CoMMA consortium, and was replaced by another partner (specialised in construction industry), we questioned the relevance of the TM scenario elicited from the first partners' practices. An implication was that we needed to adapt the scenarios for testing usefulness and usability.
- A second issue was the *adequacy of the interfaces*. If we did interact directly with end-users for specifying the generic scenarios and the system functionalities, we lost this direct relationship when specifying and implementing the CoMMA interfaces. The main reason was that the first-step priority of the project was for designers and knowledge engineers to assess the integration of the different technologies (agents, XML, ontologies, Machine Learning techniques) underlying CoMMA design. Hence, the first interfaces were actually built for designers and knowledge engineers to test the integration, and not for end-users to test the usability of the integrated technologies. These interfaces were closer to specialised interfaces of systems like Protégé or OntoKnowledge than to interfaces dedicated to technological monitors or new employees, that is to say real users. As a result, the initial interfaces did not reflect at all the expectation of the end-users; they could not have a clear view of the functionalities offered by the system and its usage. This clearly appeared during the installation of the system at the end-user sites, and during evaluation demonstrations of the system at their sites. To make functionalities visible and at hand to end-users, we were obliged to use representations more meaningful to them, mainly, stories and story-boards—two scenario-oriented representations.

To explain more precisely why the interfaces were not usable (interface evaluation), ergonomists [Giboin *et al.*, 2002] performed a cognitive walkthrough of the interfaces [Wharton *et al.*, 1994]. It allowed them to identify a number of usability criteria that were not met by the interfaces, and to propose recommendations to meet these criteria. Among these criteria were the Nielsen's usability heuristics [Nielsen, 2001a], and the Scapin and Bastien's ergonomic criteria [Scapin and Bastien, 1997]. They also performed a survey of existing interfaces which propose functionalities similar to the CoMMA functionalities, and which generally met the usability criteria emphasised by the cognitive walkthrough of the CoMMA interfaces. These recommendations served as a basis for re-designing the CoMMA interfaces. They also exploited the scenarios, because they provide a more appropriate vision of the users, of their workplace, of their tasks, and of the processes (e.g. information-seeking) they develop to achieve these tasks. As we saw for the ontology engineering, scenarios proved also useful to facilitate the communication between designers and users.

11.3.1.2 Final trial evaluation

For the first trial, the evaluation of the CoMMA interfaces was done in one-shot. Interface designers mainly built the interfaces without interacting directly with end-users. Once designed, the interfaces were evaluated with real end-users, or against actual user-oriented scenarios.

For the second trial, we also had a final evaluation of CoMMA with end-users, but this evaluation was prepared by a series of intermediate evaluations involving "users as designers" *i.e.*, end-users who participated directly to the re-design of the interfaces. In other words, we had an iterative evaluation process, consisting of a series of design-and-test cycles. The participants were: users-as-designers, GUI designers, human factors specialists, and some designers-as-users interested in the GUI re-design process (among which I was).

The final user evaluation was performed in two steps:
- During a first step, users were invited to *use the CoMMA system to perform tasks related to scenarios of use meaningful to them*. Evaluators gathered qualitative and quantitative data from these scenario executions, precisely users' comments and spontaneous recommendations about the interface and its related functionalities. The number of users involved in the evaluation for each scenario was 6 for the technology monitoring scenario and 4 for the new employee integration scenario. Users' comments were classified in terms of positive and negative usability aspects, *i.e.,* in terms of usability criteria met or not met by the interfaces. Expression of recommendations were encouraged.
- During a second step, 4 users for technology monitoring and one user for new employee integration were asked to *rate the severity of the negative usability aspects identified* by all the users during the first step, using a rating scale adapted from [Nielsen, 2001b]. During this second step, a GUI designer (Philippe Perez) was also asked to assess the design effort necessary to implement the proposed recommendations. The goal of these two ratings was to determine the importance of the critics made to the second version of the interfaces.

The second trial results clearly showed that the CoMMA system was not only still useful (its functionalities were accepted by users), but also had become usable: the GUIs being less complex, users accepted them, and were far less reluctant to manipulate them.

Concerning severity ratings of these aspects, or problems, by users, we observed that:
- Technology monitoring users considered 36,55 % of the problems as minor, cosmetic, or not a problem at all. The proportion was almost the same for the new employee integration user (35,29 %); however the set of problems placed in this category was different and this underlines the shift in the scale of importance of functionalities when changing the scenario.
- Technology monitoring users considered 39,73 % of the problems as major. However, only 17,65 % of the problems were considered as major by the new employee integration user (35,29 %). Technology monitoring users considered 17,95 % of the problems as catastrophic.

The remaining percents concerned problems not rated by the users (these problems were mostly problems that the users did not mention during the first step, and that did not relate to meaningful aspects of users' tasks).

The GUI Designer evaluated the design effort necessary to solve the problems:
- 35,30 % of the modifications to be performed were rated minor, light, or they were said to have already been done in the lapse of time that the evaluation was performed;
- 21,57 % of the modifications were rated major;
- no modification was considered as heavy;
- 2 % were rated as impossible to perform.

The GUI Designer did not rate 41,48 % of the problems because no modification (recommendation) was proposed to these problems.

One limitation of these trials was the fact that they were small-scaled evaluations: not only the number of organisation members (users) was limited, but also the number of annotated documents within the organisation, and above all the duration of system use in the organisation. If users effectively used the system, they used it on short periods of time. This "short-period use" evaluation did not allow the evaluators to observe searching, indexing, and learning phenomena that are only perceptible on longer periods of time.

Consequently a large-scaled test is needed, with more users, more documents, and more annotations. It is worth mentioning however that during an open day organised with potential end-users from different industrial and academic organisations, several persons in the public acknowledged the potentialities of the CoMMA system, and some of them asked us to apply the system, or a similar solution, to their organisations. Such collaborations could be a way towards "real-scale" testing.

Although the number of users and of evaluators was quite small, the results found by evaluators and given here are interesting indicators that I shall use to motivate some perspectives. More precisely, these results are, in my opinion, symptomatic of the gap that is forming between, on the one hand, the huge complex conceptual models underlying new symbolic artificial intelligence solutions and, on the other hand, the naturally focused conceptualisation of users in their daily tasks.

11.3.2 Criteria used for specific aspects evaluation

Before the evaluation was carried out, the CoMMA consortium brainstormed over the criteria to be used. I reproduce here in Table 31, Table 32, Table 33 and Table 34 the definitions that were chosen to guide us in this process.

Not all these criteria are relevant for a given component of the system. However just like the scenario grid guided the scenario description and data collection and analysis, these tables enabled the each designer to consider the different criteria that were relevant to evaluate the components.

Criteria are grouped in three categories: user-friendliness in Table 31, deployment in Table 32, engineering in Table 33 and efficiency in Table 34.

Criteria	Definition
Appropriateness	Degree in which the considered component or aspect represents a relevant and appropriate answer to the different issues that must be addressed by the application scenarios.
Usability	Easiness for the users to learn, manipulate, provide their input and interpret the output of the component to achieve tasks involved in the application scenarios.
User adaptivity	Capability of the component to adapt itself to the user's behaviour or to provide customisation means.
Accessibility	Capability of the component to provide immediate and natural access to the different functionalities.
Exploitability	Degree of ease with which the component could be realistically implemented.
Pro-activity	Ability of the component to take initiatives on behalf or to assist a user.
Explicability	Capability of the component to explain results or behaviours.
Versatility	Capability of the component to be used in varied scenarios and contexts.
Guidance	Capability of the component to provide assistance to the user.

Table 31 User-friendliness

Criteria	Definition
Interoperability	Degree to which the component communicates or interfaces with another one.
Portability	Ability to port the component from a system environment to another one.
Flexibility	Degree to which the component is adaptable in terms of functionality, structure and context.
Scalability	Capability of the component to be deployed on a large scale or to be used for large problems.
Reliability	Probability that component will not cause a failure for a specified time under specified conditions.
Security	Degree of security guaranteed by the system regarding access rights, data storage safety, transaction secrecy.
Cost	Capability of the component to provide cheap implementation solutions.
Integrability	Ability of the component to work and to communicate with legacy systems.

Table 32 Deployment

Criteria	Definition
Modularity	Degree to which the component is itself subdivided in separated parts, or participate to the subdivision of the system into separate and independent parts.
Maintainability	Degree to which the component is amenable to changes after it has been deployed in a real solution.
Documentation	Ability of the component to propose documentation facilities.
Reusability	Degree to which the component or parts of it could be reused with few or no adaptations.
Extensibility	Capability of the component to facilitate the addition of new functionalities.
Expressiveness	Capability of the component to offer meaningful representation to the user.
Feasibility	Degree to which the design of such a component is realistic.

Table 33 Engineering

Criteria	Definition
Time of response	Elapsed time between the submission of a request to the component and the supply of a result.
Relevancy	Ability of the component to provide adapted and meaningful results.
Completeness	Ability of the components to provide the whole set of expected answers as results.
Consistency	Ability to not provide false or contradictory results.

Table 34 Efficiency

You most probably notice that some criteria are linked together. In fact, since the evaluation was carried out by persons with different profiles (different designers in charge of different components, different users or managers, ergonomics, etc.), these links reveal relationships between aspects in different facets of the evaluation. Depending on the profile of the evaluator, the point of view adopted and the component evaluated, some criteria will be preferred to others.

In the following sections, I shall use these criteria to summarise the return on experience for each one of the components I detailed in the previous chapters. A much larger set of components was evaluated in CoMMA, but only the ones which are relevant to this work are presented here, and I invite the user to consult the deliverable reports of CoMMA for further details.

11.3.3 The ontology aspect

11.3.3.1 Appropriateness and relevancy

The ontology is a cornerstone of the system since it provides the building blocks for models, annotations and messages, with their associated semantics. As I stressed, actual keyword-based search engines such as the ones used for web searching are limited to the terminological occurrences of the extensions of concepts and the introduction of ontologies frees us from this restriction by enabling agents to reason at the intensional level.

The ontology provides the semantics needed for semantic web technology by formalising the relevant semantic aspects of the concepts used for annotating, structuring and searching the memory. When evaluating information retrieval techniques, two important indicators are precision and recall:

- *Precision* indicates the percentage of the retrieved documents that are relevant to the initial user's need. e.g. if we have retrieved 20 documents and 6 can be considered to be matching our needs, then we would say that precision of the retrieval technique on that test is of 30%.
- *Recall* indicates which percentage of relevant documents present in the base was actually retrieved. For instance, if there are 60 relevant documents and only 15 of them were retrieved the recall percentage would be 25%.

I recall here the examples given in section 6.3 showing that both precision and recall can be improved thanks to ontological knowledge such as the taxonomy of concepts:

- A query looking for *documents* concerning *vehicle* security will also retrieve *documents* concerning *car* security, *reports* on *train* security, etc. since all these concepts are specialisations of the concepts used for querying; the answers retrieved are relevant, thus the recall is improved compared to a purely term-based retrieval.
- A query looking for *documents* concerning *pipe* diameters, with pipe being an instance of the concept of "tubes", will not retrieve documents concerning the diameter of pipe as smoking tools; the precision of the query is thus improved.

Thus ontology-based annotation by structuring the memory and allowing inferences on its content, is a powerful way to increase precision and recall; in this example the ontology enables to reason at intensional level to reduce the noise (improve precision) while the use of taxonomic links enables the system to retrieve information that would not have been found if the tests were only at the 'term level' (improve recall).

11.3.3.2 Usability and exploitability

Ontology usability and exploitability can be considered from the two points of view, of the user and of the designer:

- From the *user point of view*, the ontology does improve the system behaviour and effectiveness. The system can rely on this conceptual structure to perform meaningful inferences and improve the intelligence of its symbolic manipulations. One condition, however, is the development of suited interfaces to make that conceptual structure transparent all along the use of the system. This aspect is further discussed in section 12.1 page 360.
- From the *designer point of view*, it introduces efficient coding standards and high-return and high-performance techniques. Ontology-based design does ease the integration phase by enabling the agent coupling at the high level of semantic message-passing. It makes explicit the semantic consensus between the agents and captures it in an external logical theory (the ontology object itself). However the additional design burden can be prohibitive if it is not thoroughly planned, controlled and restricted to effective needs. Tools to support the ontology management are definitely needed.

11.3.3.3 Adaptability, maintainability and flexibility

We encountered several problems with the user interface in the first trial; they led me to think that ontology-human interaction needed an intermediary layer for a terminological alignment between the user and the system. I showed that we need to explicitly represent the terms and their links with the concepts. That is why I included the terminological level inside the ontology to allow the system to take into account multilingual interfaces, ambiguity in terms, synonymous terms, etc. This should enable the system to adapt to different users, contexts and uses. One problem that I solved by this way, was the foreign language problem *i.e.,* German users wanted German terms to denote concepts while Italian users wanted Italian terms. I addressed it within the ontology itself having multi-language labels inside the ontology for each notion; the last version of CoMMA only has the English and French labels that are the only languages I can maintain.

The adaptability also means maintenance and customisation which would have to be tackled in a complete solution; this maintenance cycles on the ontology may come to be a heavy recurrent burden all the more as the ontology is the basis for other consensus and that its maintenance may lead in its wake the maintenance of the above components.

In CoMMA, a first draft of the ontology was a good step for feasibility study and first prototypes, but refining, validation and checking work is heavy and it comes with no surprise that the prototype lifecycle is time consuming. Reviews of the ontology were also triggered by feedback from trials, end-users' complaints about what was missing, or what has not been conceptualised or formalised properly. *An ontology is a living object*, the maintenance of which has consequences beyond its own lifecycle: it has an impact on everything that was built upon it. A software where the ontology was hardwired has to be versioned, knowledge base consistency has to be maintained, etc. Therefore, although the problem of the ontology evolution itself is a hard one, one should consider the fact that ontologies provide building blocks for modelling and implementation. What happens to the elements that were built thanks to these building blocks when a change occurs in the ontology? Deletion and modification obviously raise the crucial problem of consistency and correctness of the annotation bases. But an apparently innocuous addition of a concept also raises the question of the annotations using a parent concept of the new concept and that could have been more precise if the concept had existed when they were formulated. The question is: should we review them or not? These problems are obviously even more complex in the context of a distributed and heterogeneous system. The development of tools supporting ontology engineering and lifecycle is a vital condition for making maintenance and evolution realistic and therefore, for making ontology a credible software design option.

11.3.3.4 Accessibility and guidance

One of the wishes that came from the end-users was to reduce the complexity of the ontology, especially the top level that introduces some highly philosophical distinctions and some abstract concepts. The end-users expressed the wish of simplifying these levels, but the visualisation complexity should not be a reason for flattening an ontology, and these high levels are important for checking the soundness and for processes such as the query generalisation.

The best way to tackle this problem is to hide the complexity through ergonomic interfaces. An ergonomic and pedagogical representation interface is a critical factor for the adoption of the ontology by the users; if users are overloaded with details or lost in the meanders of the taxonomy, they will never use the system and the lifecycle of the ontology will never complete a loop.

We investigated this problem and developed a tool for annotation that enable users to manipulate the ontology. We also showed that XML technology and especially the XSLT stylesheets can improve the navigation and the appropriation of the ontology by users and designers; this discussion is further detailed in section 12.1 page 360.

However it is to be acknowledged that the accessibility of software relying on more an more complex conceptual structures must be studied carefully and that it places additional burden on the ergonomic design aspects of the project.

11.3.3.5 Explicability and documentation

The ontology captures the formal semantics and includes natural language to explain meaning of concepts. The first aspect enables agents to reason at the intensional level, and to disambiguate exchanges. It improves search by augmenting expressiveness of the language for annotating and querying and it also improves the precision of expressions (compared to ambiguous terms of keyword plain text search).

The final formal version of the ontology is documented by the natural language definitions, comments, remarks, that are exploited by users trying to appropriate the ontology, it is intelligible both to computers and humans. This plays an important role in documenting the querying and result interfaces making them more intelligible; for instance the concepts manipulated for building a query or a result are documented by their natural language definitions displayed in pop-up windows that facilitate the user's interpretation.

However, a missing aspect is the capture of the design rationale of the ontology, that would explain the form it currently takes and justify the choices that led to this state.

11.3.3.6 Expressiveness, relevancy, completeness, consistency and reliability

The ontology was created with several iterations and then frozen for the trials. The concepts and relations were expressive enough to be able to instantiate the needed annotations for the new employee and technology monitoring scenarios for both trials. Thus expressiveness and completeness were acceptable for the purpose of the trials.

Full, consistency checking would have required ontology tools and additional formalisation that were not available. Elementary checks are performed in CORESE such as non circular subsumption and other warnings. The top of the ontology was studied using previously mentioned theoretical work. But a lot of aspects could not be verified and the O'CoMMA ontology could most probably be improved in its completeness and consistency.

11.3.3.7 Versatility, modularity and reusability

As said in chapter 8, O'CoMMA has more or less three layers which are interesting to consider for versatility, modularity and reusability concerns:

- A very general top layer that roughly looks like other top-ontologies: it is extremely abstract and general and therefore, potentially highly reusable.
- A very large middle layer that tends to be divided into two main branches: (1) a branch generic to corporate memory domain (documents, organisation, people...); it is reusable in the context of any other corporate documentary memory and (2) a branch dedicated to the topics of the application domain (telecom: wireless technologies, network technologies...); it is potentially reusable in the context of the same application domain, but it will most probably require rework and extensions to reflect the conceptualisation of end-users.
- An extension layer which tends to be scenario and company specific with internal complex concepts (trend analysis report, new employee route card, etc.); it will change as soon as the company or the scenarios are changed.

Beware, the reusability of a part does not mean that it will not require adaptations and updates; it only means that the part can be used as a starting point for a new application in order to save time.

11.3.3.8 Extensibility

Extensibility is an intrinsic characteristic of ontologies where notions can be subsumed by more general notions and/or can be specialised by more specific concepts. The example of CSTB extension tables given in chapter 8 showed how such a process of extension can be achieved.

Extensibility concerns the three characteristics of the coverage of an ontology: exhaustivity, specificity and granularity. Exhaustivity is extended by enlarging the domain of application or the cases considered in the scenarios. Specificity is extended by detailing the identification of mobilised concepts. Granularity is extended by detailing the axiomatisation.

11.3.3.9 Interoperability and portability

The ontology provides the semantic grounding for the agent-agent, user-agent and user-user interactions. The whole annotation and querying system relies on the fact that the ontology provides a *non-ambiguous shared conceptual vocabulary* to describe the resources and express the patterns of searched information. The whole speech act theory used for the FIPA ACL (Agent Communication Language) messages exchanged is based on a shared ontology; Figure 76 page 286 showed a message using three ontologies: the FIPA ontology, CoMMA management ontology and O'CoMMA. The shared ontology is therefore, a cornerstone of interoperability in the system at the semantic level.

O'CoMMA is formalised in RDF(S) using the XML syntax and therefore, can be exchanged with and used by other software compliant with this standard, independently of the operating environment.

11.3.3.10 Feasibility, scalability and cost

Compared to the Web, a corporate memory has a more delimited and defined context, infrastructure and scope (the corporation) and thus an ontological commitment is conceivable to a certain extent. The work of building an ontology is a tough one: it is extremely time consuming and the ontological commitment needed from the users is difficult to obtain even in a small group, and becomes more difficult as the commitment concerns larger communities. The cost is especially high due to the lack of integrated development platforms for ontology engineering. Finally, the ontology evolves in prototype cycles slowly refining and ever changing. The ontology in itself can grow without major problems, but the scalability to larger projects or communities raises serious time and logistic concerns. The precision needed for the ontology and the available resources have to be considered in feasibility studies before any project is started.

We used scenarios to capture in a natural and efficient way the end-users' needs in their context. This is vital for a symbiotic integration of the system in the work environment and to focus on the specific aspects of knowledge management involved in our case, capture the whole picture and a concrete set of interaction sequences providing something to check up on every new idea, every contribution, in particular the ontology contributions. The associated data collection methods are good for completing, detecting overlooked needs, confirming or invalidating hypotheses, but they are time-consuming and cannot be envisaged on a large scale. One specific problem for the knowledge engineer is the access to the documents: for security reasons, multi-lingual context problems, etc. documents may not be available or may not be understandable. To deal with that problem, we tried to train the end-users to do the collection and analysis themselves, with guidelines. But they did not have time to do it; in fact, the problem is that we need to initiate the process, to show them the results to convince them and get them to implicate themselves. The prototype lifecycle reappears here with not only a growing refinement, but also a growing implication.

To speed up the process, I also decided to reuse existing ontologies. These ontologies have not been imported directly or translated automatically. I found that the best way to reuse them was to analyse their informal version as a natural language document that would have been collected. Divergences between the objectives, the modelling contexts and the use of these ontologies and those of our ontology (end-users and application scenarios are indeed different) made me think that no automatic import is possible and that human supervision was compulsory.

Natural language processing tools can help the analysis and translators between formal languages can ease reuse. Natural language processing tools are also of an extremely valuable help for document corpus analysis and the terminological study.

What is actually missing is a supporting tool for the methodology: a lot of good contributions to the theoretical and formal problems have been done and I have studied several of them, but to be able to manage, share and discuss the growing ontology, ontologists would definitively need an integrated environment with:
- improved interfaces for representation, navigation and manipulation of ontologies.
- tools for natural language processing, based on statistics, linguistic rules, heuristics, etc. to ease and semi-automate the process of text corpus analysis, to manipulate and exploit the extensive part of textual resources.
- facilities for applying the results from theoretical foundations of ontologies and help ontologists check their taxonomy and the ontology in general.
- tools to manage the versioning of the ontology and all that has been built upon it (e.g. issue corrective scripts for annotations, models, inferences...) and to capture the design rationale.

Most of the work done until now concentrates on some specific step of the ontology lifecycle or are prisoners of a predetermined formal language, overlooking intermediary steps and representations.

11.3.4 The semantic web and conceptual graphs

In this part I evaluate both the use of RDF(S) and its XML syntax as formalism and the CORESE API for the development of agents managing semantic web annotations and schemas.

11.3.4.1 Usability, exploitability, accessibility and cost

I formalised the ontology in RDF Schema and we wrote the annotations in RDF using the XML syntax. RDFS language was used for exchanging schemas (ontologies) on which the annotations are based. In the context of a semantic corporate web, it is interesting to use such a standard to exchange the ontology between agents and to use RDF to annotate the documents in the memory.

Inference and query mechanisms have been developed and tested, and are available to manipulate existing formalisms such as the Conceptual Graphs. Using the mapping from Conceptual Graphs to RDF, we have been able to quickly test our ideas and hypotheses on the exploitability of the semantic web formalism. Moreover, there exists a real adequacy between the two models: RDFS classes and properties smoothly map onto Conceptual Graph concept types and relation types as shown in Figure 39 page 208.

Thus the API provided by CORESE (see section 6.3) allows agents to mine the corporate memory through its annotations. The projection mechanism takes into account the hierarchies and specialisation relations described by the Conceptual Graph support obtained from the RDF schemas, thus the semantic search improves precision and recall thanks to exact typing and subsumption semantics. It also allows for tuning the matching processes, enabling approximate matching or generalisation. From an external point of view and as illustrated in Figure 41 page 210 the users of CORESE (e.g. designers of CoMMA) feed the engine with RDF(S) inputs and receive RDF(S) outputs without knowing the internal conceptual graph implementation. Additionally an inference engine enables the users to provide rules to be applied on the annotation base, in the format of XML/RDF(S) files.

The whole set of tools used for the RDF(S) and XML management (parsers, XSLT engines, etc.) is available online for free. CORESE was developed in the ACACIA research team and required several man-years of developments. Alternative engines options are multiplying and commercial solutions will soon be available.

11.3.4.2 Flexibility, adaptability, extensibility, modularity, reusability and versatility

XML is extensible in the sense that one can define new tags and attribute names to parameterise or semantically qualify data and documents. Unlike HTML, XML tags describe the structure of the data, rather than the presentation. Content structure and display format are completely independent. The eXtensible Stylesheet Language (XSL) can be used to express stylesheets, which have document manipulation capabilities beyond styling. Thus a document of the corporate memory can be viewed differently and transformed into other documents to adapt to the need and the profile of the agents and the users while being stored and transferred in a unique format. The ability to dissociate structure content and presentation enables the corporate memory documents to be used and viewed in different ways. The terminological level in RDF together with the XSLT stylesheets used for interface and documentation purposes have proved to be real assets. They allow different views for different users/contexts/uses with alternative navigation facilities.

RDF uses a simple data model expressed in XML syntax to represent properties of Web resources and their relationships. It makes no assumption about a particular application domain and thus is extremely versatile. The annotations are based on an ontology and this ontology can be described, shared and extended thanks to RDF Schema. RDF Schema is related to object models (Classes, Properties, Specialisation,…), however properties are defined independently from classes and multi-inheritance so that they can be extended by anyone.

CORESE exploits the external schemas and annotation files that capture ontological and fact knowledge. It is thus a completely generic engine also independent of domain since its implementation and inferences only rely on the RDF and RDFS recommendations.

A legacy application is a program or a group of programs in which an organisation has invested time and money and usually it cannot be changed or removed without considerable impact on the activity or the workflow. Just as an important feature of new software systems is the ability to integrate legacy systems, an important feature of a corporate memory management framework would be the ability to integrate the legacy archives, especially the existing working documents. Since RDF allows for external annotations, existing documents of the corporate memory may be kept intact (word processor document, spreadsheet, image, etc.) and annotated externally. Also the XML syntax is designed for interoperability and enables wrappers of legacy systems to generate XML that can be reintegrated in a new system such as CoMMA. However, interfacing CORESE with external sources such as databases has not been considered until now.

11.3.4.3 Expressiveness

One of the first problems encountered was the redundancy of information that may appear. For instance, annotating a document as multi-modal is redundant with the fact that it is annotated with the different modes it uses. So we decided that the multi-modal concept was not a basic concept and that it should be a 'defined concept', that is a concept derived from other existing concepts where possible. However the notion of defined concept does not exist in RDFS, and it requires an extension of the schema as proposed by in [Delteil *et al.*, 2001]. The same applies to unstable or fuzzy concepts, for instance a 'Newcomer': how long are you a newcomer in a company ? The definition could change even inside a project involving different companies. These concepts have to be defined based on another information: in our example it could be the hiring date. These choices also arise when formalising, where sometimes a notion, first formalised by a concept, can become formalised by a relation, or by an inference rule and vice-versa. For instance, the first formalisation of the notion of 'colleague' by a concept was changed into a relation. And then, considering the enterprise model, it appeared that this relation would not be used for annotation, but that it would more likely be inferred from what can be described in the state of affairs (Mr. X and Ms. Y belong to department D, therefore, there is a 'colleague' relation between X and Y).

From these first limitations and the forthcoming functionality to be implemented, the expressiveness of RDFS appears too much limited to represent the whole ontological knowledge of the corporate

memory. Axiomatic knowledge - concept formal definitions, algebraic properties of relations, domain axioms - is crucial for intelligent information retrieval on the Semantic Web and we saw that inference rules were needed to discover implicit knowledge in the annotations and enable search engines to be independent of the point of view adopted when annotating. Thus we identified the need for inferences and extended RDFS to make explicit algebraic characteristics of properties useful for inferences (transitivity / symmetry / reflexivity) and we proposed a rule language in XML/RDF and implemented the corresponding rule engine to complete the annotation base applying a forward-chaining algorithm.

11.3.4.4 Appropriateness and maintainability

The memory is, by nature, an heterogeneous and distributed information landscape facing the same problem of information retrieval and information overload than the Web. Software agents must have the ability to acquire useful semantic information from the context of the world they evolve in to quickly become intelligent actors in those spaces. Annotated information worlds are, in the actual state of the art, a quick way to make information agents smarter.

The set of elements, attributes, entities and notations that can be used within an XML document instance can optionally be formally defined in a document type definition (DTD) embedded, or referenced, within the document. The main reason to explicitly define the language is that documents can be checked to conform to it. Therefore, once a template has been issued, one can establish a common format and check whether or not the documents put in the corporate memory are valid and thus maintain its coherence and its structure. XML Schema is going to replace DTDs using an XML syntax and enabling typing of documents. Modularity is reached through namespaces to qualify the origin of tags and attributes that can be imported from virtually anywhere.

The approach of the semantic Web is extremely well suited where the semantics of documents is made explicit through metadata and annotations to guide later exploitation. XML becoming an industry standard for exchanging data, the industrial partners of CoMMA appreciated its use for building the structure of the memory. With the corporate memory becoming an annotated world, agents developed with CORESE use the semantics of the annotations and through inferences help the users exploit the corporate memory. RDF and its XML syntax allow the resources of the memory to be semantically annotated. The memory can then be considered and exploited as a semantic corporate Web.

The major issue in maintainability is that annotations depend on the ontology that provided the primitives for their structure. The problem of migrating an annotation from an old version of an ontology to a new version is still open. In the perspectives, I shall discuss the possible solution I envisaged to propagate updates of annotations with regard to the type of modification was done in the ontology.

11.3.4.5 Portability, integrability and interoperability

XML is a standard description language recommended by the World Wide Web Consortium for creating and accessing structured data and documents in text format over internet-based networks. Its simple syntax is easy to process by machine, and has the attraction of remaining understandable to humans. XML makes it possible to deliver information to agents in a form that allows automatic processing after receipt and therefore, distribute the processing load over the MAS. It is also an industry standard, and therefore, a good candidate to exchange data and build a co-operation between heterogeneous and distributed sources in a corporate memory. CORESE is not only a prototype of search engine, but it was also engineered to provide an API used in CoMMA for implementing the behaviour of the agents handling the ontology and the annotations. It is written in JAVA to be easily portable, and thus the technical capabilities of manipulating an annotation base can be included in virtually any application written in JAVA. The XML syntax of RDF and the RDF(S) model for O'CoMMA and the annotations used are also warrants of the interoperability.

11.3.4.6 Feasibility, scalability, reliability and time of response

The Notio API upon which CORESE is built provides an implementation-independent interface for manipulating Conceptual Graphs. A key feature of CORESE is the matching between a query and a target graph. The Notio graph matching operation which is quite powerful with small-sized graphs, is not usable when the target graph contains a sizeable quantity of information (e.g. more of one minute with a target graph of 50 relations). Moreover, CORESE is dedicated to reasoning upon RDF metadata translated into conceptual graphs, RDF relations being binary ones. The graph-matching algorithm of Notio has then been specialised and improved in CORESE. To summarise, four improvements have been made:
- by using heuristics to avoid a combinatorial explosion when exploring the target graph,
- by performing a connected sorting of the query graph,
- by using a cache to accelerate the process,
- by specialising the graph matching algorithm to binary conceptual graphs.

The CORESE answer time now averages out to less than one second. In particular, the new graph matching operation of CORESE enables to search upon the huge graphs of ontologies. For instance, the Gene Ontology generates 97030 RDF triples that create 10111 concept types, 53357 concept instances and 132867 relation instances and it is loaded in 37.21 seconds on a Pentium III, 1Ghz, 532 MO and biprocessor. At the time of writing this evaluation and for the query of Figure 116 with 7 arcs 4 concepts, 1 Literal constraint and 3 Literal questions, the 25 answers are found in 0,27 seconds:

```
1   <term about="?q1">
2     <name>~gene</name>
3     <isa>
4       <term about="?i">
5         <name>?n2</name>
6       </term>
7     </isa>
8     <part-of>
9       <term about="?p">
10        <name>?n1</name>
11        <isa>
12          <term about="i2">
13            <name>?n3</name>
14          </term>
15        </isa>
16      </term>
17    </part-of>
18  </term>
19
20  ------------------------------------------------------------------------------------------
21
22  [term:q1]-(name)-[Literal:~gene]
23  [term:q1]-(isa)-[term:i]
24  [term:q1]-(part-of)-[term:p]
25  [term:p]-(isa)-[term:i2]
26  [term:p]-(name)-[Literal:n1]
27  [term:i]-(name)-[Literal:n2]
28  [term:i2]-(name)-[Literal:n3]
29
```

Graph relation view

Figure 116 Testbed query

A testbed developed and used by Olivier Corby (in charge of the development of CORESE) enables him to assess performances through scenarios that provide indicators as the ones given here and ensure that the last modification did not introduce major bugs, by comparing previous results output to new ones. From the feasibility point of view, this fully operational prototype and its good results are a tangible proof of concept.

11.3.4.7 Documentation

CORESE, being developed in Java, it complies to the JavaDoc standard and therefore, comes with the complete documentation of its API in HTML format. There also exist documents on the query language and publications on the underlying principles [Corby and Faron-Zucker, 2002].

As for the semantic Web formalism, RDF(S) allows the documentation of each primitive (label and comments) thus the natural language and informal aspects have not been lost, and using XLST stylesheets, we kept the informal views at any stage of the development.

11.3.4.8 Relevancy, completeness and consistency

CORESE combines the advantages of using the standard RDF(S) language for expressing and exchanging metadata, and the query and inference mechanisms available in Conceptual Graphs formalism.

RDF(S) is the result of intensive open discussions in a workgroup of W3C and relies on experiences in knowledge modelling.

Among Artificial Intelligence knowledge representation formalisms, Conceptual Graphs are widely appreciated for being based on a strong formal model and for providing a powerful means of expression and very good readability. In CORESE they provide a sound underlying conceptual model and the available tools inherit 20 years of development experience.

The studied and published mapping between these two formalisms aims at providing the best of both worlds. However, inferences are driven by the conceptual structure of the ontology which thus influences the quality of results: inconsistencies or incompleteness of the ontology may be hidden in informal aspects the semantics of which is inaccessible to the tools (e.g. the motivation for a subsumption link) and they may introduce inconsistencies or incompleteness in the results.

11.3.4.9 Security, explicability, pro-activity and guidance

Security, explicability, pro-activity and guidance are not handled by CORESE:
- the engines does not explain its inferences and results are the only output. However some warnings or errors are issued when ill-formed schemas, annotations or queries are submitted.
- secure transactions and protection of data are not handled.
- pro-active behaviour is limited to hard coded inferences that participate to the projection algorithm and to the inference rules that are applied to the base.

11.3.5 The multi-agent system aspect

11.3.5.1 Appropriateness, modularity and exploitability

In CoMMA the multi-agent paradigm and technology were used as the enabling technology for the implementation and deployment of a distributed heterogeneous artificial intelligence system. The multi-agent platform is the software architecture of the whole system and autonomous software agents are used as coarse grained components to build the CoMMA application. In that sense, the multi-agent system was perfectly appropriate to provide a paradigm on which a solution could be envisaged and designed.

The MAS approach is intrinsically modular: the powerful agent paradigm with semantic level message coupling based on an ontology goes beyond and includes the previous paradigms (objects, components, design by contract, etc.) acknowledging autonomy, social roles and protocols as first class aspects. The agent paradigm proved its interest for software engineering and distributed implementation: agents and their behaviours could be developed by the partners of the project that had the skills and the tools needed (e.g. machine learning, semantic search engine...) and the integration was be done at the semantic level based on a shared ontology. Thus in CoMMA, agents, as loosely-coupled software components, proved to be interesting for specification, development, integration, deployment and exploitation phases. For instance, since the agents are loosely coupled software components and since their role and interactions have been specified using a consensual ontology, the integration and set up of a first prototype was achieved in less than two days and the replacement of a partner by another one was another excellent test that showed that changes required (ontology, interface) were completely contained. This separation of concerns in the system heavily relies on the ontology and communication languages (FIPA ACL and RDF).

Focusing on the annotation distribution management through agents, the trials showed an effective specialisation of the content of the annotation archives. One important point underlined by the first results is that the choice of the specialisation of the archives content must be very well studied to avoid unwanted unbalanced archives. This study could be done together with the knowledge engineering analysis carried out for the ontology building. It would also be interesting to extend the pseudo-distances to take into account the number of triples present in the archives to balance their sizes when choosing among close bids. We witnessed a noticeable reduction of the number of messages exchanged for query solving (compared to a simple multicast) while enabling fragmented results to be found. The new algorithm exploiting additional heuristics and decomposition techniques is being studied to further reduce this number of messages exchanged for solving.

11.3.5.2 Usability, feasibility, explicability, guidance and documentation

JADE [Bellifemine *et al.*, 2001] provides a complete API to develop a multi-agent system compliant with the FIPA standard, fully playing its role of a development framework. Moreover, a user-friendly graphical interface to visualise the configuration, monitor the deployment and debug the multi-agent system, thus JADE also plays its role of platform.

Even if groups and roles are not first citizens in the JADE platform, an organisational approach proved to be successful to specify the implementation. Several methodologies now propose formal models to support a sound organisational analysis. It would be interesting to compare how these different models would represent the CoMMA architecture and capture the design rationale.

"Agentisation" was difficult for large and complex components (as the solver), but this would disappear if fully finalised Agent-Oriented languages were used instead of object-oriented languages with an Agent API. Indeed JADE is not an agent-oriented programming language, there are no language primitives for agents defined in the language itself; this is normal as the agent paradigm is still under construction and the development of a complete and portable language represents an important investment. JADE is a JAVA API and a JAVA-implemented platform, thus the underlying programming paradigm is still the object one which sometimes leads to complex object manipulations to obtain what should be a basic functionality in an agent-oriented language fully implementing the paradigm.

However the training provided by the University of Parma, the debugger tools, the tutorials and guides available, the Javadoc documentation of the API, and the excellent support mailing list playing the role of help desk makes JADE a very accessible development framework. In addition, JADE provides specific libraries and factories of FIPA communication protocols and behaviours to make easier the programming of the communicative acts and to reinforce the relationship between the action and the communication aspects. JADE is a very good tool to make proofs of concepts as it was done in CoMMA.

Due to the ontology component, the FIPA ACL messages that agents exchange are somewhat self-explanatory for a developer and propagate the error messages back to the requester enabling to understand what went wrong. However, from the end-user point of view, there is a conflict between the will to hide complexity of MAS and the need to explain a problem when it occurs. In CoMMA no explication mechanism was implemented and if the return of an error could easily be displayed in the interface, it would most probably be of no use to the common run of users. It seems important to introduce and use the organisational model here too, in order to provide the system with awareness of its environment and enable it to send the right message (e.g. network failure) to the right person (e.g. network administrator).

I also showed that a methodology could be followed to obtain the architectural design of a system, following a top-down organisational decomposition and refinement of functionalities. The use of documentation standards (roles cards, extension of UML diagrams, etc.). These informal documentation enable communication between designers that were sometimes beginners in the field of agents and they guided the implementation.

11.3.5.3 Adaptability and pro-activity

Pro-activity and adaptability are created at agent-level and system-level. The push is due to a chain reaction between several roles, while the result sorting is due to machine learning algorithm in the User Profile Manager. Separation of concerns between architectural design and configuration deployment is also a source of adaptability, to the specificity of the place where the system is rolled out. The modularity of multi-agent systems is an intrinsic characteristic that gives them a natural ability to adaptability and flexibility. However much more adaptability and pro-activity could be envisioned in the system (e.g. monitoring emergence of communities of interest, animated characters and personal assistants, collaborative filtering, etc.)

11.3.5.4 Interoperability, expressiveness and portability

The interoperability requirement is fulfilled by the JADE infrastructure, that complies with the FIPA standard. The portability requirement is no concern, since the whole CoMMA system, its libraries and JADE are implemented in Java. The expressiveness could seem a strange criteria for the multi-agent system aspect, but the FIPA compliant platforms come with the FIPA Agent Communication Language. In CoMMA the expressiveness of the FIPA ACL was perfectly sufficient for the tasks to be performed; moreover, the free language for the content slot of messages enabled us to use RDF(S) as previously discussed.

11.3.5.5 Scalability, reliability and time of response

The trials were not large enough to seriously assess scalability. Scalability involves the whole integrated system, but the multi-agent system contributes to it by its distributed nature: using adequate configurations, clusters of related agents can be deployed together to improve performance.

Reliability is not easily assessed and formal proofs have not been done. The only contribution in that direction is a reuse and adaptation of established engineering methodologies where possible.

It is also possible to replicate agents in order to achieve application-level fault tolerance. Two agents engaged in a conversation do not depend on the other agents for this interaction, so that if some other part of the system crashes, they are not affected. The yellow pages managed by Directory Facilitators allow dynamic matchmaking, so there is no need for hardwired dependencies among agents. Of course, if the Directory facilitator or the Agent Management System go down, the whole system will eventually go down. Any agent can be stopped or restarted independently and at run-time.

The efficiency parameter that is the most affected by the multi-agent implementation is the response time. Although JADE limits as much as possible the marshalling, serialisation, parsing, etc. the semantic level message coupling is undoubtedly an additional burden and time-consuming task in agents. In the configuration shown in Figure 80 page 293 and used for the CoMMA Open Day, query time was a few tens of seconds. The improvements in query distribution and the specialisation of the bases tend to reduce this problem and in general the middle agent approach in multi-agent systems aims at reducing the number of messages. Still the communication time will augment with the number of relevant interlocutors identified.

11.3.5.6 Security

The JADE provides no security at all: the current framework is a single user environment and message passing is not secured. It must be stated that security concerns were not at all in the objectives of CoMMA and that the only development made in that area was the login and password identification at the beginning of each session. A support for multiple users and security was announced by the JADE team for the near future.

11.3.5.7 Cost

The JADE middleware is an Open Source project, distributed under the LGPL license and available free of charge, so no cost is associated with acquiring the MAS infrastructure.

The development cost to build the CoMMA application is surely reduced by the availability of the JADE framework, but this must not occult the fact that JADE itself results of some man-years of R&D. Lesser maturity of agent paradigm and technology at large with respect to more established and older approaches resulted in additional costs for developers training, methodology building and choices, prototype and test investigations. The development of the agent architecture was one of the biggest part of the time-plan of CoMMA.

11.3.5.8 Integrability and maintainability

As a matter of fact, the technique of multi-agent programming supports itself the integration aspect of the project; as agents are individual and active components, that need to communicate to be able to carry out the different tasks they are involved in, particular mechanisms have been recommended and implemented and rely on the shared ontology. As noticed by our partners of the university of Parma, the strong analogy between the way of programming based on agents and ontologies and the "programming by contract" explains why the integration of the different agents has been achieved quite easily. The consensus of designers themselves was at the agent role, communication protocol and semantic message-passing level too.

Legacy systems integration is eased by the multi-agent approach using wrapper or transducer techniques, thus helping integration. JADE relies on CORBA, and the whole CoMMA system is implemented in Java and uses XML; these points are assets for integrability. Finally, tools for the deployment and debug of the multi-agent system are available in JADE and the CoMMA system can be configured, observed and administered during normal operation by means these tools.

11.3.5.9 Flexibility, versatility, extensibility and reusability

Of course, the multi-agent paradigm, by nature, brings a lot of versatility and flexibility to the system. The flexibility and legacy systems integration of the CoMMA system were demonstrated during the Trial 1, when the CoMMA application was deployed at end users' sites, according to their specific network topology and document bases. The separation of concerns present in the multi-agent architecture helps reuse. Agents, ontologies and behaviours can be provided and reused independently. Still large-scaled reuse remains one of the chimeras that has been experienced in other main stream paradigm. Reusability in multi-agent systems, as in other domains, tends to be restricted to small groups of designers and developers building and reusing their in-house library.

11.3.5.10 Relevancy, completeness, consistency

These points can be applied to the distributed solving and allocation algorithms. Relevancy of allocation was only tested by comparing on some tens of annotations the choices made by the agents and the one that I would have done. Concerning the distributed solving, it is based on CORESE, thus relevancy and consistency are ensured. However, completeness remains to be proved and only some elementary mathematical characteristics of the distance were detailed. In particular, the multi-instantiation allowed by RDF poses problem that I shall detail in the chapter on short-term improvements.

11.4 Conclusion and discussion on open problems

The ontology component represents the keystone of the CoMMA system as the guarantor of the semantic grounding a semantic web. It is extensible, quite reusable or rather recyclable. On the other side of the coin, the ontology design is time-consuming. The maintenance is largely overlooked in the current state of the art and it has consequences on the whole system. Most of the proposed systems are academic prototype or extremely focused tools specialising on one task or one formalism. Several expectations have to be fulfilled before one can envisage to be able to manage, share and discuss the a living ontology in real life systems; an ontologist definitively needs an integrated environment and initiatives like WebODE (a workbench for ontological engineering [Arpírez *et al.*, 2001]) should be pursued to obtain frameworks integrating:

- user-friendly interfaces for representation, navigation and manipulation of ontologies in a co-operative context.
- tools for natural language processing, based on statistics, linguistic rules, heuristics, etc. to ease and semi-automate the process of text corpus analysis, to manipulate and exploit the extensive part of textual resources.
- tools for analysing legacy systems (databases and their schemas, internal web, etc.).
- functionalities to produce, maintain and manage the evolution of intermediary representations.
- facilities for applying the results from theoretical foundations of ontologies and for helping ontologists check their ontology.
- tools to manage the versioning of the ontology and of what has been built upon it (e.g. issuing corrective scripts for bases of facts, models, inference rules, etc.) and to capture the design rationale.
- tools to manage inter-ontology relations and manipulation of multiple ontologies: extensions, dependencies, merging.

Ontologies are definitely a powerful conceptual tool, but the complexity of their design and maintenance makes it compulsory to develop complete workshop software to assist ontologists at each stage and transition of the construction and maintenance cycles enabling the actual research results to scale-up to the scope of a real company application.

Moreover, the interfaces between the ontology and the user are a major area of concern. RDF(S) coupled with XML stylesheets provides support to deal with ontologies browsing, but the gap between the abstract conceptual models and the daily concrete concerns is making the interface designers do the splits to maintain ergonomic qualities. Ontologies should include the concepts mobilised for their own use in the envisioned application scenarios to support smarter inferences inside the interfaces that rely on them.

The RDF(S) expressively is limited by the absence of a logical layer of inference layer. Adding inference mechanisms was necessary to compensate the RDF problem of expressiveness. Bachimont [2000] decomposes the ontology modelling process in three stages, corresponding to three commitments: the semantic commitment specifying the linguistic meaning of concepts; the ontological commitment specifying their formal meaning; the computational commitment specifying their effective computer-based use. Much work is needed to help explicit, represent and preserve the intensional semantic structure of the computational level. Since the ontology is motivated by an intended use, this use has to be made explicit and it will be of great help in the ontology recycling and reusing process. If the new generation of AI agents is to be based on an explicit conceptualisation, this must not be limited to the knowledge exchanged currently, it must include the action inferences performed, with both their *intension* and *intention*.

The integration task was considered as one of the major risk factors at the very beginning of the project, but the multi-agent technology and the semantic-level coupling played their role perfectly. Integration through agents also enabled us to couple techniques from really different domains, which is usually quite difficult. A methodology inspired by organisational approaches was proposed, but methodologies for multi-agent systems suffer, as in other domains, a gap between applicable and widely understood, loosely formal approaches and constraining, but rigorous and checkable formal specifications. Our methodology was followed and feasibility was proved by practice and running prototypes, but it was clear that, as the system grew, the complexity of design documentation and the size of specifications were becoming unmanageable. Much work remains to be done in standardising methods, design rationales and notations and to integrate them in development environment supporting the user from specification to deployment and maintenance.

The fact of being an integration architecture places a lot of expectations on the multi-agent platforms (security, scalability, performances, different internal agent architecture, etc.) which makes it very difficult to provide generic platforms all the more as the paradigm itself is still searching for a stable identity. At this time, standards are both seductive (to start to build upon) and dangerous (restricting perspectives).

From the end-users point of view, the final system was both a real proof of concept and a demonstrator. It is not a commercial tool, but it did play its role in diffusing research results and convincing new partners to consider the multi-agent solution for distributed knowledge-based systems. From the developer point of view, the ontology-oriented and agent-oriented approach was appreciated because it supported specification and distribution of implementation while smoothing the integration phase. The modularity of the multi-agent system was appreciated both at development and trial time. During the development, the loosely-coupled nature of the agents enabled us to integrate changes in specifications and contain their repercussions. Moreover, industrial partners wrote in their final report, that for them the exploitable results of the projects were: new products and tools providing innovative knowledge management solutions, new services for knowledge management, and above all a gain of expertise on new approaches from transfer of knowledge between academic and industrial partners.

Thus, CoMMA was a convincing experience to show that an approach based on knowledge engineering (formalising knowledge about resources of an intraweb through semantic annotations based on an ontology) and distributed artificial intelligence (multi-agent information system loosely coupled by a co-operation based on semantic message exchanges) can provide a powerful paradigm to solve complex distributed problems such as organisational memory management.

Both the ontology and agent fields suffer from the paradox of trying to develop at more and more abstract levels while hiding this underlying complexity from the user. Explicit means of focalising and presenting this complexity will have to be taken into account at the very heart of these new paradigms. This does not hamper the fact that they are a very promising and complementary couple of paradigms. Finally, my experience with RDF(S), largely criticised in the knowledge modelling community for its lack of expressiveness, is that its simplicity is an asset for exporting the underlying concepts outside the research community; I wonder if, just like the RDF(S) framework is a lightweight knowledge modelling language, lightweight agent platforms applying the KISS motto of W3C (*i.e.,* "keep it simple and stupid") could be a good move to export this paradigm more easily.

12 Short-term improvements

One never notices what has been done;
one can only see what remains to be done.
— Marie Curie

This chapter describes beginnings of answers to some of the problems raised in the previous chapter. The ideas presented here are extensions of the work I presented, some of them already started to be implemented and tested. It is a shallow overview of some of the current work and of some non implemented ideas of short-term improvements of the existing CoMMA functionalities. The first section will look at the gap between internal system conceptual concerns and users' concerns in querying and annotating through ontologies. The two following sections will propose improvements of the pseudo-semantic distance for allocating annotation and of the distributed query solving algorithm. Finally, the two last sections respectively consider possible improvements in the services offered by the ontology-dedicated society and the annotation-dedicated society.

12.1 Querying and annotating through ontologies: from conceptual concerns to users' concerns

In order to illustrate this idea, I remind what an annotation looks like when expressed in RDF by giving in Figure 117 an example of annotation about a short movie presenting "Mediation", a project of unified interface access to heterogeneous catalogues.

```
1   <rdf:RDF
2    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3    xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#"
4    xmlns:CoMMA="http://www.inria.fr/acacia/comma#">
5
6    <CoMMA:Presentation rdf:about="http://fapollo:8080/comma/doc/mediation.mov">
7     <CoMMA:Title>Presentation of the purposes of Mediation</CoMMA:Title>
8     <CoMMA:HasForPerceptionMode><CoMMA:VisualPerceptionMode/></CoMMA:HasForPerceptionMode>
9     <CoMMA:HasForPerceptionMode><CoMMA:AudioPerceptionMode/></CoMMA:HasForPerceptionMode>
10    <CoMMA:HasForRepresentationSystem><CoMMA:French/></CoMMA:HasForRepresentationSystem>
11    <CoMMA:HasForRepresentationSystem>
12       <CoMMA:TalkingMovieRepresentation/>
13    </CoMMA:HasForRepresentationSystem>
14    <CoMMA:Target>
15     <CoMMA:OrganizationalEntity>
16      <CoMMA:IsInterestedBy><CoMMA:KnowledgeManagementTopic/></CoMMA:IsInterestedBy>
17     </CoMMA:OrganizationalEntity>
18    </CoMMA:Target>
19   </CoMMA:Presentation>
20  </rdf:RDF>
```

Figure 117 Example of annotation based on O'CoMMA

This annotation expresses that the resource `http://fapollo:8080/comma/doc/mediation.mov` is a presentation with the title "`Presentation of the purposes of Mediation`", it uses visual and audio perception. It is a talking movie in French and it is especially targeted to organisational entities (persons or groups) interested in knowledge management.

Then, Figure 118 shows a query pattern that could match such an annotation sincs it looks for titles of documents targeted to entities interested in the 'Knowledge Engineering Topic'.

```
1   <CoMMA:Document CoMMA:Designation='?'>
2    <CoMMA:Title>?</CoMMA:Title>
3    <CoMMA:Target>
4     <CoMMA:InterestAbleEntity>
5      <CoMMA:IsInterestedBy><CoMMA:KnowledgeEngineeringTopic/></CoMMA:IsInterestedBy>
6     </CoMMA:InterestAbleEntity>
7    </CoMMA:Target>
8   </CoMMA:Document>
```

Figure 118 Example of query based on O'CoMMA

From the above examples, it should be clear that there is a vital need for interfaces offering ergonomic views to bridge the gap between the users' concerns level and conceptual structures level. Interfaces such as the one currently used for development purposes (see Figure 119) cannot be handled as such by end users.

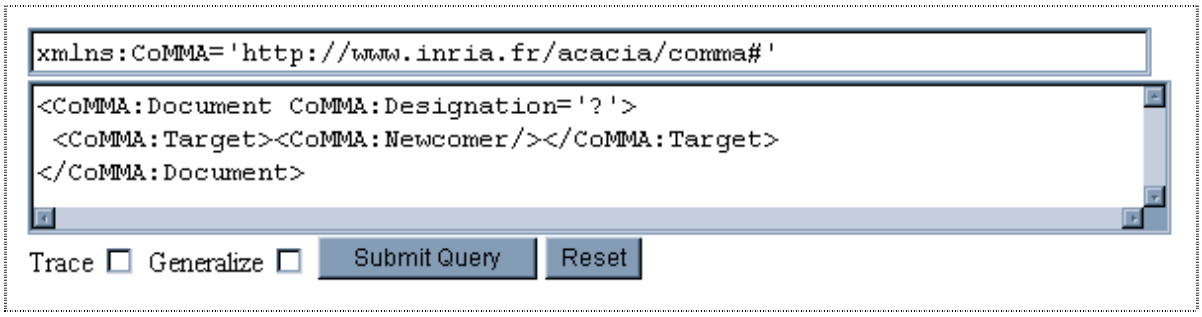


Figure 119 Tester query interface

Industrial partners confronted to the current ontology explained that it is too much complex hence abstruse to people. Especially the upper part of the ontology introduces philosophical distinctions extremely interesting from a modelling point of view, but extremely abstruse and usually useless for typical users of the system.

Moreover, the higher you are in the taxonomy, the more difficult the agreement is. Two colleagues will more easily agree on the modelling of concepts they manipulate and exchange in their daily work (e.g. a news is a type of document presenting new information) than on the top of the ontology that requires a higher level of abstraction and deals with concepts that we are not used to discuss every day (e.g. things are divided between entities and situations, entities being things capable of playing a role in a situation) - except for philosophers maybe.

The top-level deals with cultural and even personal beliefs. Its concepts are useful for system internal manipulations (e.g. generalisation of queries, structuring of higher layers), but not for the direct interaction with users. An ergonomic and pedagogical representation interface is a critical factor for the adoption of the ontology by the users; if the users are overloaded with details or lost in the meanders of the taxonomy, they will never use the system and the lifecycle of the ontology will never complete a loop.

In ACACIA we are currently studying different types of interfaces to annotate and query. And we must admit that results from the first prototypes show that the ergonomics issues are far from being solved.

The first example given below shows the interface developed in APROBATIOM assisting stakeholders of a project in the domain of construction. As shown in Figure 120, the interface is purely in HTML. The left part enables the user to navigate in the ontology to pick concepts or relations. The right part enables the user to build the query and displays help on possible actions.

Figure 120 Query interface in APROBATIOM [implemented by A. Ginepro and A.P. Manine]

We introduced a terminological level in O'CoMMA in order to explicitly capture and manage the relations between notions and terms that may be used to refer to them. It is an asset that should be thoroughly exploited when interfacing the user with the system. The first possibility based on this level is a keyword-fashioned interface where the user enters terms that are translated into a list of concepts (if a correspondence can be found) used for querying. This is the first mode of the terminological interface shown in Figure 121.

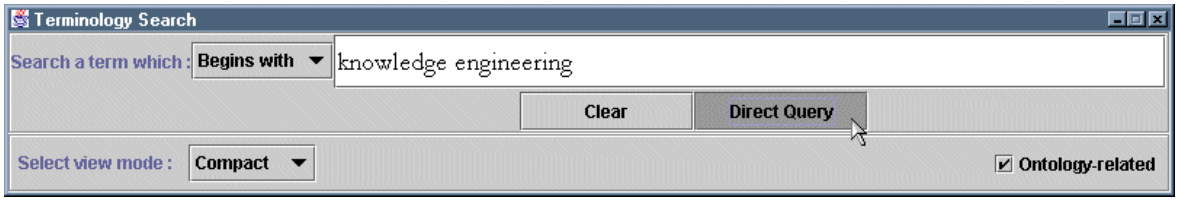


Figure 121 Simple mode of terminological interface [implemented by J. Guillaume]

While the previous interface is simple, it does not really exploit term-notion links to disambiguate the dialogue between the user and the system. Therefore, in a more complex mode, the interface proposed by Alain Giboin, enables the users to 'make shopping' in the notions available to them in the ontology; the list of notions being derived from the terms or just the first characters the user typed. Figure 122 shows this interface using the now natural Web metaphor of the trolley / shopping cart. The users virtually drag and drop the notions relevant to the document they are looking for into the shopping basket and submit it when all the notions they are looking for are inside.

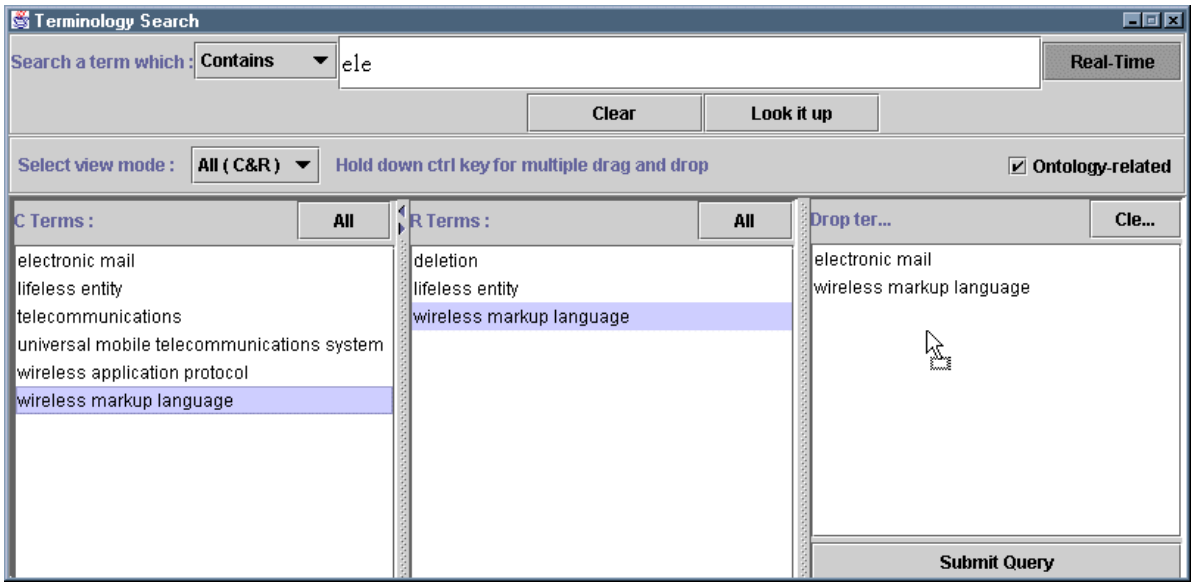


Figure 122 Extended mode of terminological interface [implemented by J. Guillaume]

Some of the notions are natural concepts, others are relations. In the previous interface, they are treated alike. This interface does not allow the user to structure the query or constrain some fields. This is the purpose of the expert interface shown in Figure 123. However I find interesting this intermediate step that is perfectly sufficient for some queries or some users.

In Figure 123, the user is building a query asking for annotation talking about sports event in "Le Mans" and Renault Cars with an engine dated 1995. This last interface is very powerful, but as a drawback, it is much more complex to use.

CoMMA interface is a merging of the terminological interface with this one: notions found are used as building blocks for query elicitation, thus saving the user from browsing the ontology (see Figure 79 page 290).

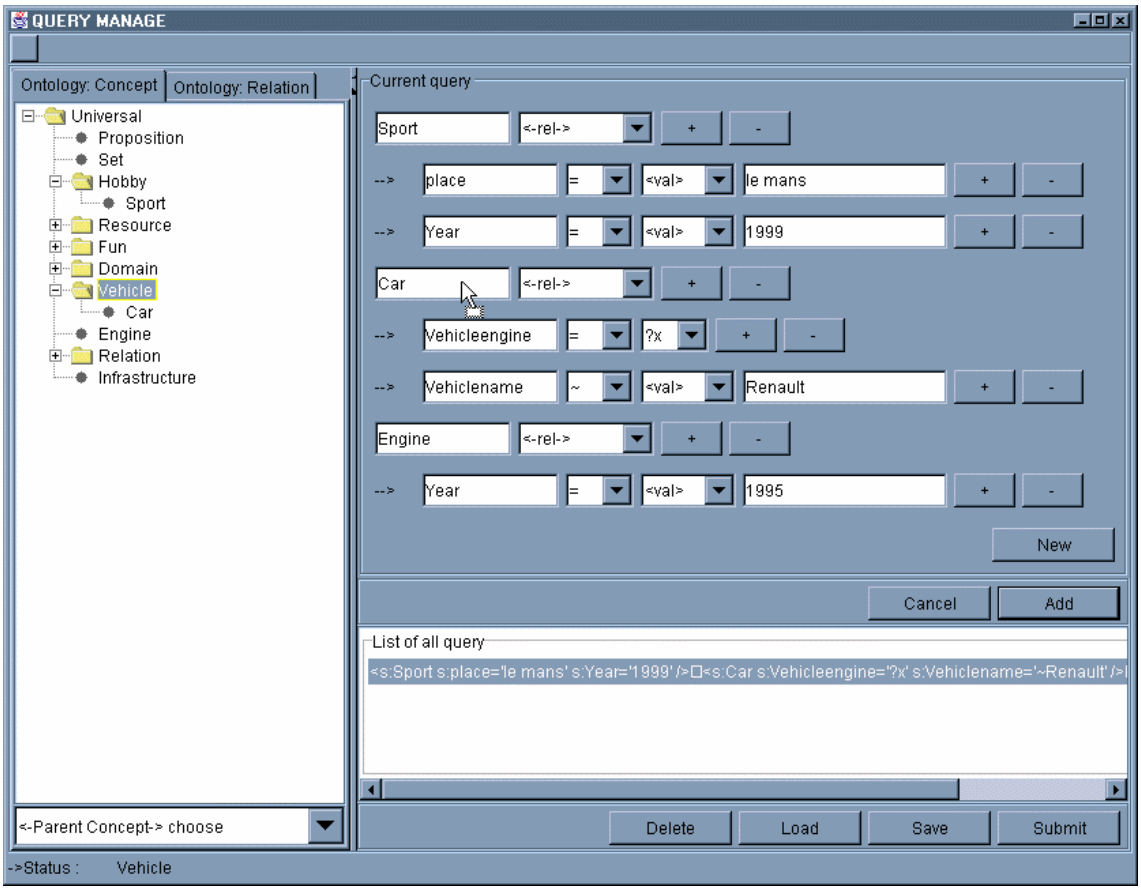


Figure 123 Expert query interface [implemented by Phuc Nguyen]

To these interfaces we must add the ones developed to browse and query the ontology itself. Ontologists are a special type of users that require a special type of interfaces. (Snapshots are given in 0 page 232).

Moreover, the ontology and its terminology can also be used to present and document the results from a query. Starting back from the query of Figure 119 asking for documents targeted at newcomers, the results given by the systems are:

- documented, for instance Figure 124 shows the use of natural language definitions in the ontology to disambiguate the term organisation chart
- presented in English by extracting terms from the ontology as shown in Figure 125,
- presented in French (Figure 126) with the same ontology, but a different terminological level. It could also be used to adapt to the user's jargon.

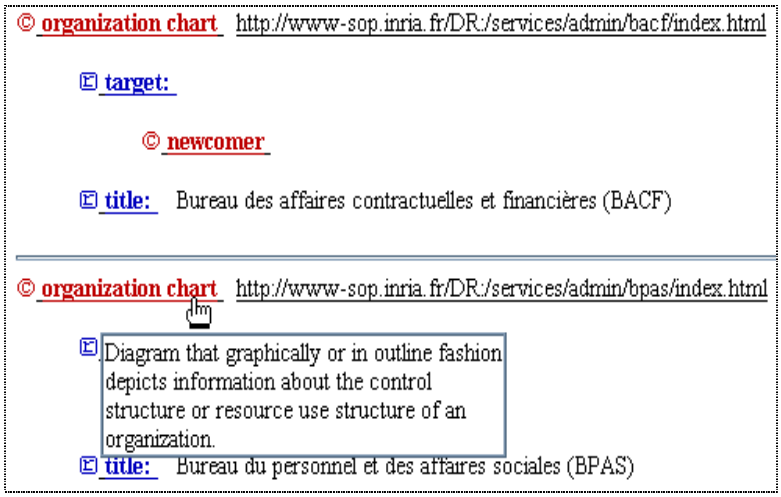


Figure 124 Documenting results

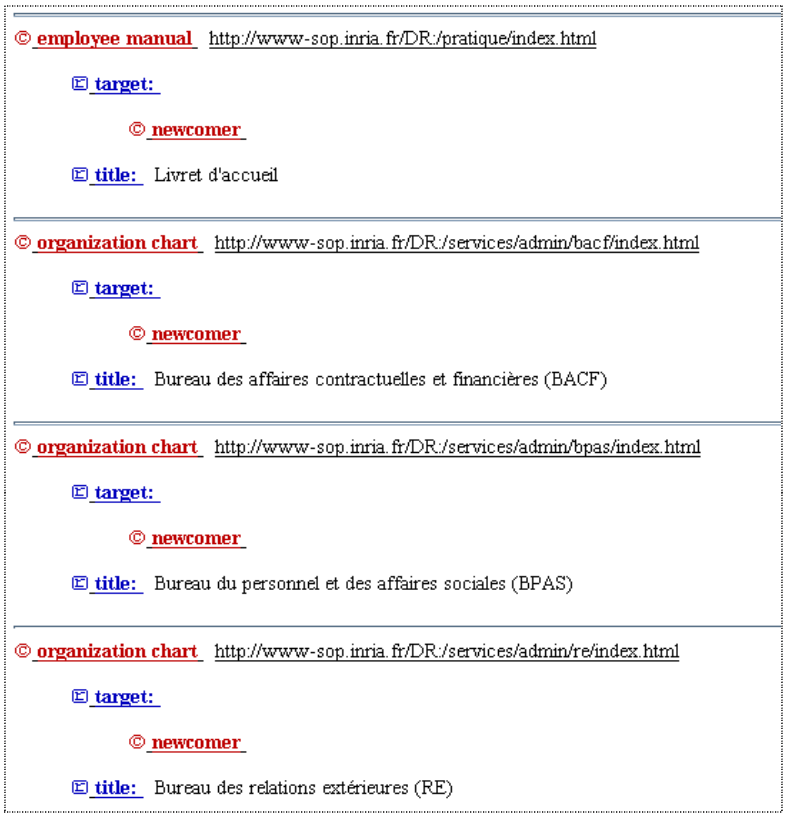


Figure 125 Result displayed in English

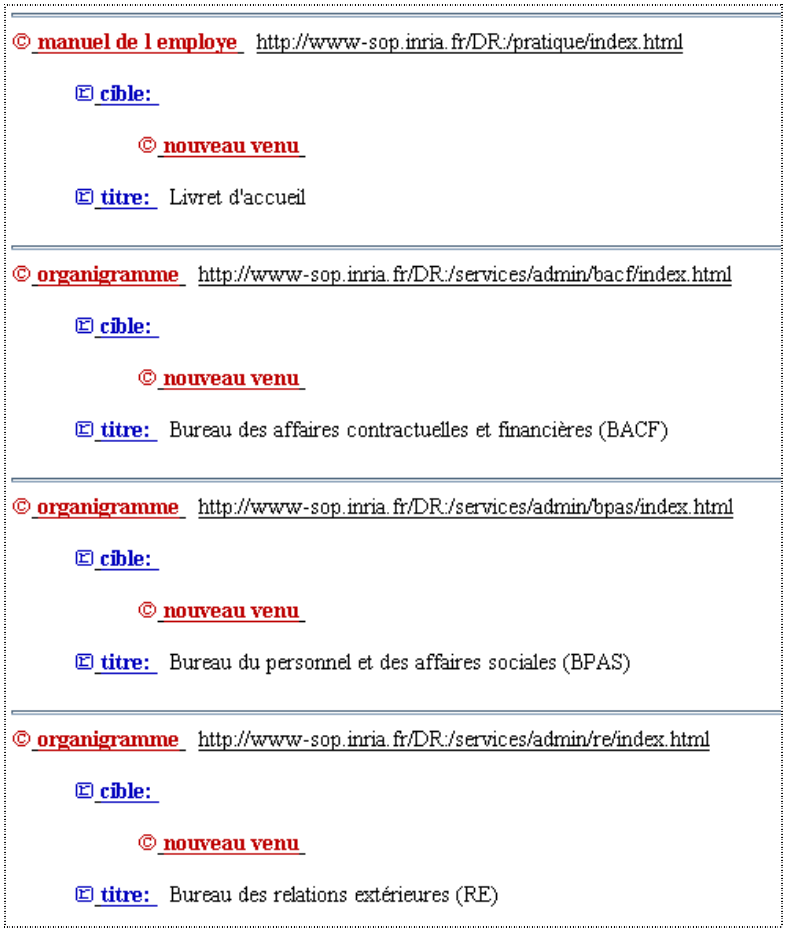


Figure 126 Result displayed in French

The ontology relevant aspects are not the same depending on the stakeholder and the context of use. Therefore, it makes sense to envisage to use the user profile to filter the access to the ontology and hide the complexity which is still visible in the previous interfaces.

In CoMMA, I tested an interface using preferred entrance points to the ontology, recorded in the user profile, to propose middle concepts (e.g. person, document, organisation) from which the user can start navigate the ontology. Just as web directories (e.g. Yahoo) constitute an alternative to search engines on the web (e.g. Altavista, Google), the I envisaged directories to the memory, the navigation being based on the ontology and the indexing being based on the annotations. Figure 127 shows an extract from my user profile with some entrance points in the ontology. These entrance points personalise the browsing of the ontology by allowing us to customise the middle concepts for each user.

```
1   <CoMMA:Employee rdf:ID = "http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/">
2
3     <CoMMA:FamilyName>Gandon</CoMMA:FamilyName>
4     <CoMMA:FirstName>Fabien</CoMMA:FirstName>
5     <CoMMA:HireDate>1999-11-02</CoMMA:HireDate>
6
7     <CoMMA:HasForWorkInterest>
8       <CoMMA:ComputerScienceTopic/>
9     </CoMMA:HasForWorkInterest>
10
11    <CoMMA:HasForPersonnalInterest>
12     <CoMMA:MusicTopic/>
13    </CoMMA:HasForPersonnalInterest>
14
15    <CoMMA:HasForOntologicalEntrancePoint>
16     <CoMMA:ComputerScienceTopic/>
17    </CoMMA:HasForOntologicalEntrancePoint>
18
19    <CoMMA:HasForOntologicalEntrancePoint>
20     <CoMMA:Service/>
21    </CoMMA:HasForOntologicalEntrancePoint>
22
23    <CoMMA:HasForOntologicalEntrancePoint>
24     <CoMMA:Document/>
25    </CoMMA:HasForOntologicalEntrancePoint>
26
27    <CoMMA:HasForOntologicalEntrancePoint>
28     <CoMMA:Person/>
29    </CoMMA:HasForOntologicalEntrancePoint>
30
31    <CoMMA:HasForOntologicalEntrancePoint>
32     <CoMMA:OrganizationGroup/>
33    </CoMMA:HasForOntologicalEntrancePoint>
34
35  </CoMMA:Employee>
```

Figure 127 Ontological entrance points in a user profile

Then, a stylesheet generates a view of the ontology for a given profile (Figure 128). Therefore, the users will not see the top abstract concepts and can start to browse the ontology from concepts that are familiar to them (Figure 129) refining or broadening notions (Figure 130). Once a given notion is selected, the system automatically generates short queries among which the user can choose (Figure 131). The results are the indexed documents for which an annotation exists and matches the short query as shown in Figure 132.

Machine learning techniques could be envisaged here to identify frequently used concepts in order to improve initial filtering, navigation and result presentation. Preferred term denote a notion could also be learned to ease disambiguation.

Figure 128 Customised directory view of the ontology

Figure 129 Entering the ontology

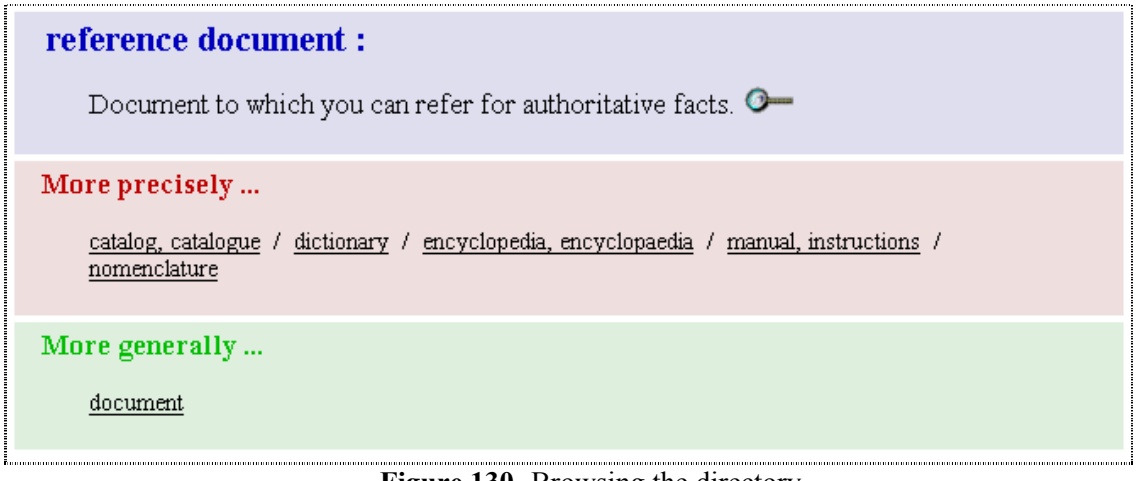


Figure 130 Browsing the directory

Figure 131 Automatically generated simple queries

Figure 132 Indexed document as a result

These interfaces illustrate the variety of options that can exist. Of course, natural language analysis interfaces (*i.e.,* where one can type a query in natural language) would be very popular and versatile. But I believe that depending on the task (searching, browsing, strolling, flâneuring, discovering, etc.), the accessibility constraints, the profile of the user, some types of interfaces and customisations may be more efficient than others.

十人十色

ten people, ten colours[1]

— Japanese saying

[1] Everyone has their own tastes

12.2 Improving the pseudo-distance

I showed how some aspects of the underlying graph model of the semantic Web framework could be exploited to handle allocation of annotations through a contract-net protocol in a multi-agent system. I discuss here some foreseeable improvements to the pseudo-distance used for bidding.

12.2.1 Introducing other criteria

The first tests on the prototype implemented showed an effective specialisation of the content of the annotation archives with the problem that the choice of the specialisation of the archives content must be very well studied to avoid unwanted unbalanced archives.

I suggested that this study could be done together with the knowledge engineering analysis carried out for the ontology building. However automatic assistance could also be envisaged in particular to take into account additional criteria and heuristics:

- If two archives are close with regard to their content, then the system could try to *balance their storage volume and workloads*. One criterion would be to consider all the archives within a margin from the best bid (e.g. the first 10%) as good candidates and choose the one with the smallest storage and workload.
- *Monitoring tools* pointing to archives very close with regard to their content (a situation which may be created on purpose to augment archiving space) or to archives with storage volume and workload much higher than others, etc. could be interesting for managing the annotation bases.
- Finally, the measure can use *different constructors to combine sub-measures*. In the literature we can find Min(...), Max(...), Average(...), Sum(...), etc. It would be interesting to test these different types of distances and study their repercussions on the allocation of annotations.

In distributed databases, overnight automatic reorganisation of distributed bases have been studied [Özsu and Valduriez, 1999]; this solution could be extended to distributed annotation archives, however it must be noticed that (a) one of our motivation for distributed knowledge based system was to leave knowledge where it was (wrappers, locally managed data, etc.), (b) complexity of automatic reorganising is known to be prohibitive.

12.2.2 The non-semantic part problem

A major drawback of the current distance is the part which is not semantic *i.e.,* the lexicographical distance between literal values. [Maedche and Staab, 2001] used the edition distance of Levenshtein [Levenshtein, 1966], that measures the minimum number of token insertions, deletions, and substitutions required to transform one string into another. It is more robust than the lexicographical distance I used, but it suffers from the same problem of not being semantic at all.

In fact, this part of the measure encounters the same problem as in information retrieval, so some ideas of this field could be reused:

- *TF*IDF approach*: to remove stop words and build vectors of the literal values on one side and of each notion of the ontology on the other side, using the labels (term and synonyms) and the natural language definition. Then, a cosine measure or equivalent could be used to choose the closest notion(s) and use it/them in evaluating the distance.
- *Label mining*: to mine the literal value to extract labels of concepts in literal; this could require to couple O'CoMMA with a general thesaurus such as Wordnet[1] to get both domain specific notions

[1] http://www.cogsci.princeton.edu/~wn/

and general domain knowledge to avoid lack of common knowledge. Then, a graph distance could be used again.

- *Natural language processing*: to produce conceptual structures from the literal and consider the whole annotation once each literal has been replaced.

12.2.3 The subsumption link is not a unitary length

Humans are an apes who knew how
to climb in the tree of knowledge,
but what distance did they climb ?

— Me ☺

The Latin proverb *natura non facit saltum* says the nature does not make jumps, which I interpret as a plea for the research of continuums between binary extremes. The subsumption link is considered by the Least Common Super Type measure as a universally unitary link *i.e.,* a subsumption link always represent a path with a length of 1. Is this intuitively natural? Consider the taxonomy in Figure 133: the unique length of 1 for each subsumption shows how the current distance consider the taxonomic tree when measuring it.

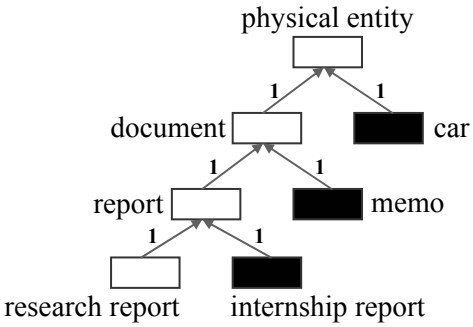


Figure 133 Taxonomy with unitary subsumption links

The first question is: "is it intuitively normal that the length of the link between a *car* and a *physical entity* be the same as the length of the link between a *report* and a *research report* ?" In my opinion the answer is already no, but the second question makes it even more obvious: "is it intuitively normal that the calculated distance of the path from *memo* to *internship report* be 3 *i.e.,* the same distance of the path from *memo* to *car*?" So the distance I used, shows here some limitations in capturing the semantic proximity and it comes from the fact that the length of the subsumption link *i.e.,* the intuitive semantic proximity between the subsumer and the subsumee is not homogeneous in the taxonomy of the ontology.

A first idea is that since the upper layer is more abstract, the subsumption distance could be attenuated with the depth of the notion as shown in Figure 134.

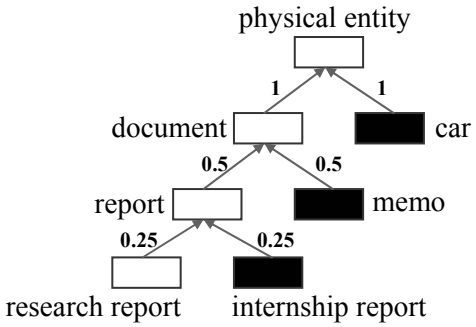


Figure 134 Taxonomy subsumption link distances dependent on depth

Here the length of the subsumption link is given by:

$$Length(Subsumption(type_1, type_2)) = \left[\frac{1}{2}\right]^{depth(type_1)} \quad (38)$$

with *depth(type)* a recursive series defined by:

$$\begin{cases} \text{depth}(\top)=0 \quad \text{with } \top \text{ the root of a taxonomy} \\ \text{depth(type)} = 1 + \underset{s_t ; Subsumption(s_t, type)}{Min} depth(s_t) \end{cases} \quad (39)$$

Using this the path from *memo* to *internship report* has a distance of 1.25 while the path from *memo* to *car* has a distance of 2.5 .

A use of this distance is to propose a non binary projection *i.e.,* a similarity function that, for instance, takes values in [0,1] where 1 is the perfect match and 0 the absolute mismatch. The initial idea comes from [Sowa, 1984] who applied it to similarity measures in conceptual graphs allowing sideways travel in the lattice of the ontology. [Ralescu and Fadlalla, 1990] applied it to the join operation on graphs. More recently [Zhong *et al.*, 2002] used an equivalent distance to build a similarity between two conceptual graphs an carry out semantic search. This is much better since the distance no longer considers the subsumption link as having a constant length, however we replaced this hypothesis by a new one which is "the length of the subsumption links decreases with the taxonomic depth". This means that the taxonomy must be homogeneous in its choices of differentia at each and every level. To illustrate that point, let us consider Figure 135.

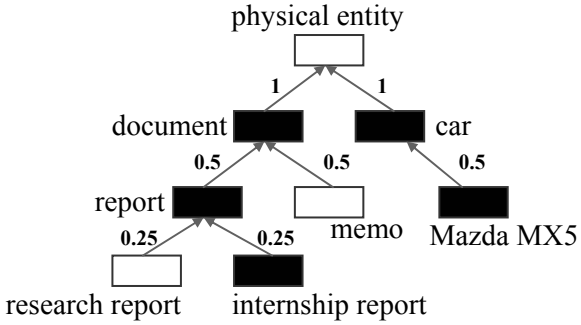


Figure 135 Taxonomy with non homogeneous differentiation

The distance between *report* and *document* is 0.5 *i.e.,* the same than between *car* and *Mazda MX5* while the distance between *report* and *internship report* is only 0.25. It is clear that the difference between *car* and *Mazda MX5* is more precise than between *report* and *document*, and so the concepts should be more distant. Thus what I would like to express is rather what is shown in Figure 136.

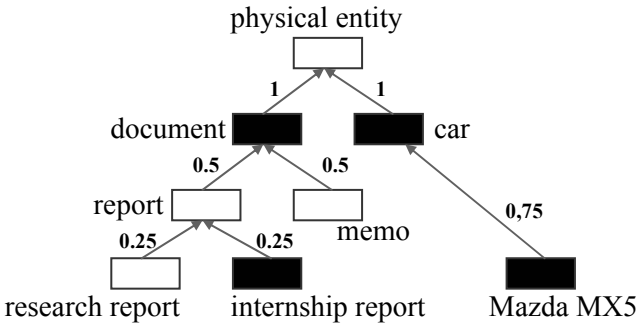


Figure 136 Taxonomy with adjusted distances

Here the important difference between *car* and *Mazda MX5* is captured in a longer link having a length of 0.75 just like the path from document to internship report.

To summarise these remarks, I used the original schema of differentia and semantic axis of [Kassel *et al.*, 2000] to explain this subsumption distance as illustrated in Figure 137.

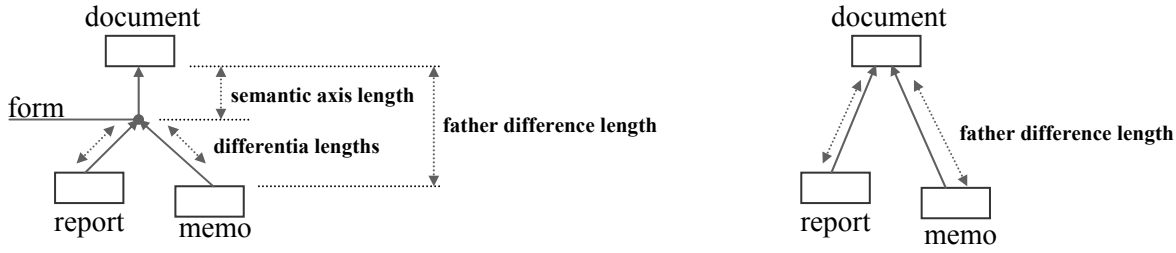


Figure 137 Semantic axis, differentia and subsumption distance

As a first remark, this distance of a subsumption link has the following characteristics:
- in an ontology, a taxonomic length of zero means no differentia:

$$length(notion_1,notion_2)=0 \Leftrightarrow notion_1.NecessaryConditions = notion_2.NecessaryConditions$$

A symptom of this characteristic is that *notion₁* and *notion₂* are their own least common subsumer, and a Formal Concept Analysis would place them at the same node.
- conversely, in an ontology, an infinite taxonomic length means no inference can go through this path:

$$length(notion_1,notion_2)=\infty \Leftrightarrow notion_1.NecessaryConditions \cap notion_2.NecessaryConditions \equiv \varnothing$$

A symptom is the absence of least common subsumer for *notion₁* and *notion₂* and the taxonomy they belong too is at least divided into two disconnected graphs to which *notion₁* and *notion₂* respectively belong.

The left part of Figure 137 proposes to distinguish the semantic axis contribution to the distance from the contribution of the differentia values:
- the semantic axis groups the children of a notion according to the characteristics involved in the definition of their differentia. The semantic axis reifies the brother community and difference principles. The choice of the axis can influence this distance, for instance to choose the *form* or *use* of the document as an axis is intuitively much less precise than to choose the *average size in number of characters*.
- the differentia values taken for each sub concept is of course also a factor of influence for the distance: the *"report"* value for the *form* is a much less precise difference than *new employee route card*.

The precision of the differentia value is influenced by the choice of a semantic axis, but can still vary a lot. For instance, the axis *average size in number of characters* could give birth to two different sets of sub types:
- one not too precise and restraining: *short document* (< 1000 characters) *vs. long document* (≥1000)
- one much more precise: *brief* [1,100]; *small*]100,1000]: *medium*]1000,10000]; *long*]10000, 100000]; *huge*]100000, ∞[.

These remarks made me identify another problem: intuitively the distance between *huge* and *brief* is bigger than between *brief* and *small*. However here, the path between two brothers is given by the distance of the path through the father. This concern reminds me of the principle of brother difference and community [Bachimont, 2000], and thus led me to envisage to differentiate the distance to the father from the distance to the brothers.

So as shown in Figure 138, there exist two influences (the semantic axis and the differentia values) and two semantic distances (the distance to the father and the distance to the brother)

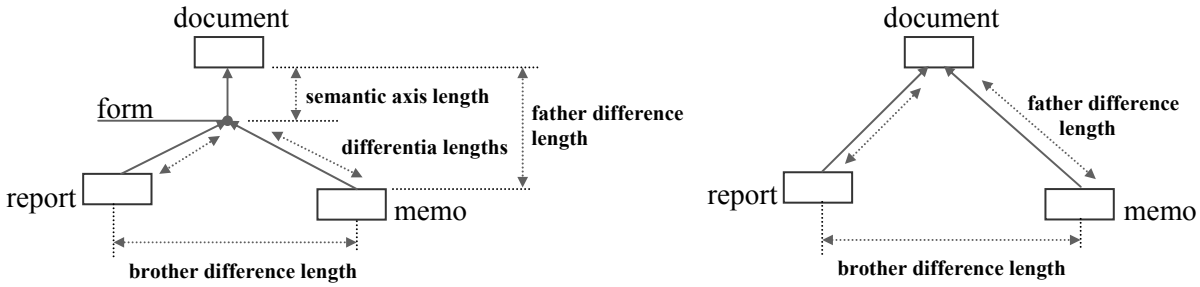


Figure 138 Father and brother distance

Thus the new distance $Dist_{Sem}(T_{A0}, T_{B0})$ between two types T_{A0} and T_{B0} could be defined by:

Let T-T' iff T is a direct super-class of T'

Let $T>T'$ iff $(T - T') OR (\exists T'' ; T\text{-}T'' AND T''>T')$

Let T_{A0} and T_{B0} be two types

Then if we $T_{A0} \equiv T_{B0}$ then $Dist_{Sem}(T_{A0}, T_{B0}) := 0$

Else if $T_{A0} > T_{B0}$ then $Dist_{Sem}(T_{A0}, T_{B0}) := Min(\Sigma Length(T_{Bi}, T_{Bi+1})$

with T_{Bi} such that $T_{A0}\text{-}T_{Bn}\text{-}...\text{-}T_{B2}\text{-}T_{B1}\text{-}T_{B0}$

Else if $T_{B0} > T_{A0}$ then $Dist_{Sem}(T_{A0}, T_{B0}) := Min(\Sigma Length(T_{Ai}, T_{Ai+1})$

with T_{Ai} such that $T_{B0}\text{-}T_{An}\text{-}...\text{-}T_{A2}\text{-}T_{A1}\text{-}T_{A0}$

Else if $\{ Type\ T;\ T> T_{A0}\ and\ T> T_{B0} \} \equiv \varnothing$ then $Dist_{Sem}(T_{A0}, T_{B0}) := +\infty$

Else $Dist_{Sem}(T_{A0}, T_{B0}) := Min\ (Min(\Sigma Length(T_{Ai}, T_{Ai+1})) + length(T_{An}, T_{Bn}) + Min(\Sigma\ length(T_{Bi}, T_{Bi+1})))$
$T_{LCST} \in \{ Type\ T;\ T> T_{A0}\ and\ T> T_{B0} \}$

with T_{Ai} such that $\boldsymbol{T_{LCST}}\text{-}T_{An}\text{-}...\text{-}T_{A2}\text{-}T_{A1}\text{-}T_{A0}$

and T_{Bi} such that $\boldsymbol{T_{LCST}}\text{-}T_{Bn}\text{-}...\text{-}T_{B2}\text{-}T_{B1}\text{-}T_{B0}$

While this ability to tune the distance is interesting, the work of "evaluating the intuitive distance" and annotating the taxonomy is clearly difficult and time-consuming, and most probably extremely subjective. Thus I would suggest to initialise the taxonomy with an automatic system such as the previous top down algorithm based on depth and the inter-brother lengths could be initialised with the sum of the distances to the father. Then, adaptation and learning techniques could be used to tune the taxonomy using usage feedback.

One usage where feedback could be relevant is the generalisation process. To define a domain of generalisation for a query in the space of queries could be equivalent to fix the length of individual generalisation per instance of notions in the query or to give a cumulated length for the sum of generalisations of instances of notions in the query. Then, generalisation suggestion could increment allowed generalisation per the average length of subsumption links in the ontology, or let the user choose in a sorted list of candidate generalisations, or each request for generalisation would look for the next closest generalisation and apply it. From this, we can capture feedback from the user about the suggestion and adapt the lengths in the ontology. Feedback could be about proposed generalisations (the closer ones are the ones chosen) or from results of the generalisations (the closer ones are the ones that generated appreciated answers).

These distances could also be used in maintenance of the ontology. For instance a growing length *can* be an indicator of a missing intermediary step in generalisation. The system could sort the biggest gaps, and compare to the average length in the ontology or the average length of the brother notions to suggest a missing step as illustrated in Figure 139 where the left part indicates a missing step that is added in the right part.

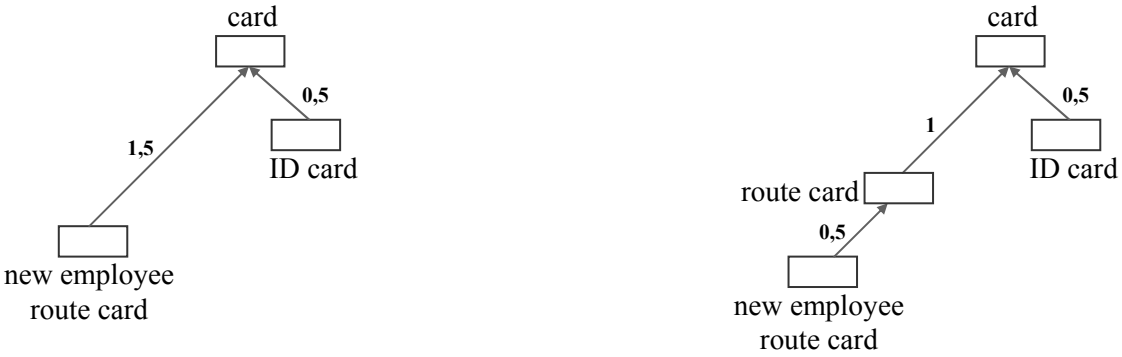


Figure 139 Indicating missing subsumption steps

Comparing the average distance of each notion with its brothers to the average distance in the brotherhood would be an indicator of homogeneity of the brotherhood differentiation.

Of course, all these ideas are only possible perspectives of improvements.

12.3 Improving distributed solving

I explained how the description of the notions used in a base could be exploited to allocate tasks in distributed query solving. I give here some improvements that could be envisaged for this mechanism.

12.3.1 Multi-instantiation and distributed solving

Multi-instantiation in RDF allows one object to be described from several points of view. However, when a query requires knowledge from different facets, this may cause a loss of information in the current distributed algorithm. Consider the two annotations in Figure 140 and Figure 141 that represent two different facets of myself.

```
1   <CoMMA:Researcher rdf:ID = "http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/">
2    <CoMMA:FamilyName>Gandon</CoMMA:FamilyName>
3    <CoMMA:FirstName>Fabien</CoMMA:FirstName>
4    <CoMMA:IsInterestedBy> <CoMMA:ComputerScienceTopic/> </CoMMA:IsInterestedBy>
5   </CoMMA:Researcher>
```

Figure 140 Researcher facet of Fabien Gandon

```
1   <CoMMA:Lecturer rdf:ID = "http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/">
2    <CoMMA:IsInterestedBy> <CoMMA:MusicTopic/> </CoMMA:IsInterestedBy>
3   </CoMMA:Lecturer>
```

Figure 141 Lecturer facet of Fabien Gandon

Annotation in Figure 140 expressed I am a Researcher and I am interested in computer science, while annotation in Figure 141 expressed I am a Lecturer and I am interested in music. If you consider that these two views should not be merged and that for instance I should not be considered as a Lecturer interested in computer science, then everything is fine. Conversely, if you believe my interest are attached to me and not to a facet of me then there is a problem: consider the query in Figure 142, it looks for lecturers interested in computer science, with their name and first name.

```
1   <CoMMA:Lecturer>
2    <CoMMA:FamilyName>?name</CoMMA:FamilyName>
3    <CoMMA:FirstName>?firstname</CoMMA:FirstName>
4    <CoMMA:IsInterestedBy> <CoMMA:ComputerScienceTopic/> </CoMMA:IsInterestedBy>
5   </CoMMA:Lecturer>
```

Figure 142 Query on merged facets

If annotations in Figure 140 and Figure 141 are gathered in one annotation, CORESE will have no problem for solving the merge between the two instances. If they are in different annotations or even on different sites, then the distributed algorithm will fail because the following constraint is false on both annotations taken separately: **Lecturer — Is Interested By — Computer Science Topic**. Indeed, the projection on the first annotation will fail because *Lecturer* does not match *Researcher* and the projection on the second annotation will fail too, because *Computer Science Topic* does not match *Music Topic*. A solution to remedy to that problem is to decompose the constraint into two constraints solved one after the other:

- the *relation using its generic signature* and the available URIs if they have been solved by previous decomposition steps.
- the *typing constraint* using the previously retrieved URIs.

This generic idea should be studied together with the use of the ABIS to improve the solving.

12.3.2 Constraint sorting

The new constraint solving algorithm, looks for the constraint focal point *i.e.,* the concept the most constrained in the query to start with. This heuristic tends to rapidly reduce the number of candidates at each stage of the solving. Additional heuristics and improvements could be proposed; two of them are given here:

- *use the nature of the constraint*: some constraints are intuitively strong and more precise than others. The current algorithm starts with URI because the universal identification is the strongest identification constraint. Otherwise if no URI has been found, it uses literal constraints to try to solve a URI and to use it as constraint afterward. This could be generalise by classifying the existing constraints and using the classification to find at every stage of the query solving the best next move to do. Such a sorting of constraint could look like what is given in Table 35; an equivalent system is already used in the centralised version of CORESE.

Rank	Constraint nature
1	One URI is known
2	Several URI are possible
3	a literal property is know ('=' operator)
4	a literal property value must contain a given string ('~' operator of string inclusion)
5	a literal property value has an ordering constraint ('<' or '>' operators)
6	a non option property question is attached

Table 35 Ranking constraints

- *use the statistics on the base*: the ABIS contains the population size for each relation, an additional heuristic could be to use the frequency of relation in the base and choose the less frequent to cut down number of possibilities as soon as possible.

12.3.3 URI in ABIS

Finally, as the ABIS describes the range of literal values, it could be interesting to find a mechanism to describe the set of genuine URIs (not the one generated for existential quantification).

The full list of URIs may be too long, but compression techniques could be used such as regrouping the URI with a common root together. This could also be used to merge multi-instantiation annotations, thus avoiding the first problem evoked in this section. The description could be used in annotation distribution as an additional clustering feature.

I find interesting to study this point in details to decide if it can be done and, if so, how it can be done without generating too much messages or disturbing the semantic specialisation of the archives.

12.4 Ontology service improvements

The current ontology service in CoMMA is extremely limited since it only covers the case of a request for a full download of the ontology. I describe here two conceivable improvements of this service.

12.4.1 Precise querying to retrieve the ontology

One can notice that different agents do not need to access the same aspects of the ontology for their respective works. A first level of difference can be found for instance between Annotation Archivist, the Annotation Mediator and the Interface Controller:
- The Annotation Archivist and the Annotation Mediator need the intensional formal structure (subsumption, signature of relations) and nothing more.
- The Interface Controller needs the intensional formal structure as well as the natural language labels and the natural language comments in a given language (French, English, etc.) for interface purposes.

A possible improvement would be for the agents to formulate a query expressing exactly what aspects of the ontology they are interested in. For instance the Annotation Archivist and the Annotation Mediator would generate a query such as the one in Figure 143 whereas the Interface Controller would generate a query such as the one in Figure 144.

```
1   <rdfs:Class rdf:ID="?ID">
2    <rdfs:subClassOf rdf:resource="?parentID $option"/>
3   </rdfs:Class>
```

Figure 143 Request full taxonomy of concepts

```
1   <rdfs:Class rdf:ID="?ID">
2    <rdfs:comment xml:lang="en">?definition</rdfs:comment>
3    <rdfs:label xml:lang="en">?term</rdfs:label>
4    <rdfs:subClassOf rdf:resource="?parentID $option"/>
5   </rdfs:Class>
```

Figure 144 Request full taxonomy of concepts with natural language information in English

In reply, the ontology agent would only provide them with the part they need. The use of precise query to retrieve the ontology from an ontology agent could be easily done in CoMMA.

This could be extended to download only parts of the ontology that are relevant to an agent.

12.4.2 Propagating updates of the ontology

In a complete system, it would be important to organise the ontology agent society to manage and propagate updates of the ontology. A simple distinction between different types of updates is:
- Addition of a new concept or new property: this could be handled easily.
- Addition or correction of terminological level (terms and definition); that raises no major problem.
- Major edition (deletion, modification of taxonomic links,...); it would require much more work and would have to be coupled with coherence maintenance in the bases of annotations.

The modification of the schema could benefit from XSLT techniques: an XSLT script could be issued to update a definition, add a term, modify a range/domain etc. The agent should also propose subscription to other agents that may want to be notified of any changes occurring in the ontology such as the Annotation Archivist, the Annotation Mediator, the Interface Controller, etc.

12.5 Annotation management improvements

Finally, the annotation management service could be extended too with additional services; the first one would be "nice to have" while the two others would be vital in a complete solution.

12.5.1 Query-based push registration

In the current system, push is triggered by a mere registration for "new annotation" events; once it has registered to the Annotation Mediator, the User Profile Manager receives a copy of every new annotation. An improvement idea would be to use the society of the annotation mediators to match the push mode requirements and make a first filter before pushing them.

The registration would be based on a query to determine if a new annotation is interesting for a given subscriber. The pushed annotation could then be filtered early in the process. For instance, an agent working on users' profiles may want to be notified of any new document concerning knowledge engineering. In this purpose, it would issue a request for registration, containing a query like the one in Figure 145.

```
1   <CoMMA:Document>
2    <CoMMA:Designation>?d<CoMMA:Designation>
3    <CoMMA:Concern>
4      <CoMMA:KnowledgeEngineeringTopic />
5    </CoMMA:Concern>
6   </CoMMA:Document>
```

Figure 145 Example of a possible registration query

Thus the push mode would be much more customisable and would generate much less messages.

12.5.2 Allowing edition of knowledge

In the actual core of the RDF(S) search engine CORESE, nothing enables the modification/edition of existing annotations. From the CORESE point of view, the ability to modify an annotation requires to additional functionalities:
- the ability to identify an annotation (unique identification); to be able to point the annotation.
- the ability to remove an annotation, or replace it by a modified version.

While these functionalities could be easily added to CORESE, serious additional problem appear when considering the same problem in a distributed environment and a complete solution:
- the ability to identify an annotation is not sufficient, since an annotation resulting from query solving may be the result of a merging of several annotations, thus it is necessary to keep trace of the origin of the different parts of a generated annotation so as to be able to trace the modification back to the source. Needless to say that any result of a calculation could not be easily modified; the same problem exists in the database field and extensions of proposed solutions could be studied.
- the second problem comes from the management of rights, because in a complete solution edition and deletion rights would have to be managed to ensure a minimum security.

12.5.3 Update scripting and propagation

When a modification of annotation is issued, it could be a very precise modification (e.g. correct my family name currently spelled "gardon" by the new spelling "gandon") or a very generic corrective script (e.g. the web site changed its root, so change every URI starting with "www.atos.com" by the same URI starting with"www.atos-origin.com").

Here again, the system could use XSLT to propagate update scripts in the annotation bases, the update protocol would be:

1. An agent in charge of interfacing the human administrator with the system would send a request to run the script to all Annotation Mediators.
2. Each Annotation Mediator propagates the Request to run the script to its Annotation Archivists.
3. Each Annotation Archivist sends an 'agree' if it is OK to its Annotation Mediator or 'refuse' if it is not concerned by the update.
4. The Annotation Mediators send their 'agree' back to agent in charge of interfacing the administrator if they have at least one Annotation Archivist that agreed; otherwise they send a 'refuse' to express they are not concerned by the update.
5. Each Annotation Archivist that is OK prepares its base and sends an 'inform' to its Annotation Mediator to explain if it succeeded or not.
6. The Annotation Mediators send their 'inform' back to agent in charge of interfacing the administrator.
7. If everything is OK, the administrator agent sends a 'Commit' else it sends a 'Rollback'.
8. Each Annotation Mediator propagates the 'Commit' or 'Rollback' to its Annotation Archivists.
9. The Annotation Archivists switch their base ('Commit') or cancel ('Rollback') and send an 'inform'.
10. The Annotation Mediators send an 'inform' to the administrator agent.

Inside an Annotation Archivist:

1. For each file in the directory of the base the Annotation Archivist applies the script and name the result file <SOURCE_FILENAME>.UPDATED
2. If an error occurs, the Annotation Archivist sends an 'inform' ERROR else send an 'inform' OK.
3. If a 'Commit' comes back: it renames old files <SOURCE_FILENAME>.OLD; removes extension .UPDATED of new files; and deletes old files. This way, we ensure that even if the system crashes in the middle of the process an acceptable coherent state can be recovered. Finally, it reloads the base and sends an 'inform' DONE
4. If a 'Rollback' comes back: it deletes <SOURCE_FILENAME>.UPDATED

The user requesting the update could create a query to isolate the target of the update, then in the obtained annotation, the user selects a part to modify and gives the modification to be done. From this, the system could generate a modification script starting from the closest tag having an ID in order to create an anchor for the script (this meets the previous problem of identification).

This new feature would mean that we would have to tackle the problem of the bases being off-line when an update occurs. This problem is known in Distributed-Databases to be extremely complex.

12.6 Conclusion on short-term perspectives

This concludes the description of short-term perspectives and current work. I showed that the current architecture and solutions of CoMMA could be improved in a rather seamless way *i.e.,* the implemented architecture does provide the modularity we were looking for.

I described some experiences in designing interfaces that bridge the gap between the conceptual design concerns and the day-to-day users' concerns.

Then, I proposed three improvements of the semantic pseudo-distance, which respectively address:
- the problem of introducing new criteria in the allocation mechanisms to improve archives management.
- the problem of the non semantic literal distance that could be solved by an ontology-guided mining of the literal values to come back to taxonomy-based distances.
- the problem of the subsumption link improperly considered as having a constant unitary length.

Likewise, I gave a number of possible improvements for the distributed query-solving mechanism:
- I explained the issue of multi-instantiation in distributed query solving and proposed a decomposition of the relation signature constraint in two separate constraints to tackle it.
- I proposed to go further into constraint sorting to augment the probability of reducing the communication load as much as possible.
- I evoked a possible improvement of the ABIS structure to include the URIs and take them into account when deciding if an Archivist is competent for the solving of a sub-query.

The fourth section proposed two additional services in the ontology-dedicated society: the use of precise querying to retrieve ontological knowledge; the propagation of updates of the ontology.

Finally, I envisaged three additional functionalities concerning annotation management:
- the use of a query in order to specify the registration for the notification of the submission of a new annotation.
- the required improvements to allow the edition of existing annotations.
- the use of scripts to propagate updates over the annotation repositories.

These suggestions of improvements are all directly linked to the current state of implementation and I believe that each one of them could be done in a reasonably short time (at most six man-months). On the contrary, the following chapter will envisage long-term perspectives; some of them could justify a whole project in themselves.

13 Long-term perspectives

Imagination is more important than knowledge.
Knowledge is limited.
Imagination encircles the world.
— A. Einstein

This chapter presents some considerations on the extensions that could be imagined and on needs to evolve towards a complete real-world solution. Its sections provide more questions than they give answers: therefore, I classified them as long-term perspectives. They must be read as interest statements and by no means as realistic specifications.

The first section concerns ontology and multi-agent systems and will focus on the problems of ontological lifecycle and the relations between ontologies and multi-agent systems. The second section deals with organisational memories and multi-agent systems, it first considers the need to open the system, acknowledging the fact that no actor of our scenarios is an island. Then, it will consider the dual problem of closing the system for security and system management concerns. Then, problems of distributed rule bases and rule engines will be underlined as well as the need for support of workflows and information flow automation. The following section will present some new noises appearing in these new information retrieval systems. Finally, I shall conclude with some of my dreams for the future.

13.1 Ontology and multi-agent systems

13.1.1 Ontological lifecycle

Although this point was also discussed in the specification meetings of CoMMA, it is only reported here, as it is far more complex than what was envisaged and as it would require a whole project in itself. As shown in Figure 9 page 73, the lifecycle of an ontology is a complex process. I can identify a lot of problems to be addressed in the lifecycle and complete management support of an ontology. I only give here the identity and some clues I would consider for each one of them.

13.1.1.1 Emergence and management of the ontological consensus

Much work is needed to support the emergence of the ontological consensus and semantic reconciliation in a community. The Computer Supported Collaborative Work community could provide a lot of ideas in this area. I noticed a very interesting initiative from [Mark *et al.*, 2002] to reconcile different perspectives and support communication of people who are distributed. Their Reconciler systems is a Web-based system designed to aid communicating partners in developing and using shared meaning of terms. This idea could be extended to include results from the ontology engineering society, for instance work from tools developed by [Guarino and Welty, 2002] or [Troncy and Isaac, 2002]. There are interesting challenges in studying social rules and developing assisting tools of collaborative engineering of ontologies, including:

- management of intermediary representations as defined in [Gómez *et al.*, 1996] for each stage of the design.
- ontological design principles and tools that assist the transition from an intermediary representation to the next one.
- use of bootstrapping ontology and semi-automatic analysis tools (natural language processing).
- management of social protocols and rationale to reach a consensus: e-mail notifications of changes or requests, comments through votes and surveys, history of decisions, etc.

To allow this, the ontology must include the primitives that will sustain its design rationale and intermediary stages.

Apart from the collaborative aspect of ontology engineering, a special point of interest to me here is the ability to follow the lifecycle through a set of intermediary representations and transformations. As illustrated in Figure 146, I believe XML and XML Schema could be used to propose and develop a set of documents or views corresponding to different stages of maturity or different intermediary representations. Based on these representations, transformations (may be using XSLT) could be defined to assist the ontologist in moving along the formalisation dimension. In my opinion, this would make a difference with the majority of tools concentrating on one task and the ability to manipulate and review the intermediary productions would ease maintenance and revision of the ontology. As we saw in CoMMA, these views are also interesting documents in themselves, providing lexicons of the organisational structure, infrastructure, domains, activities, etc.

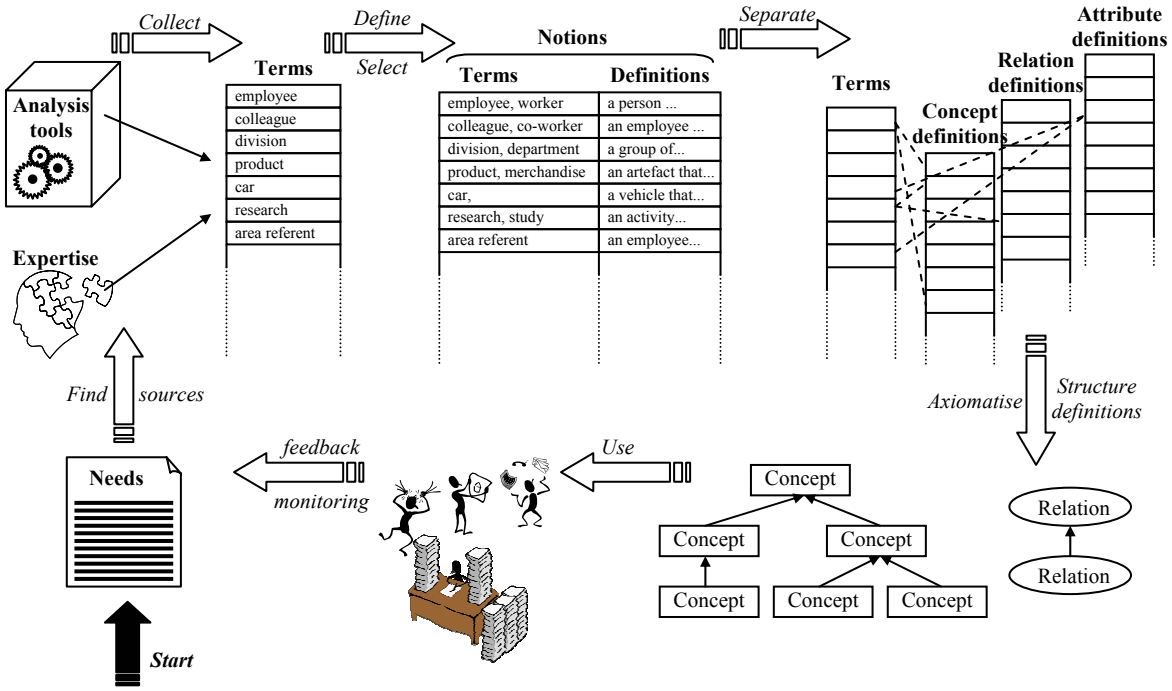


Figure 146 Ontology cycle and intermediary stages

13.1.1.2 Maintenance and evolution

The evolution of the ontology is partly linked to its emergence and design: some tasks are the same. However, additional constraints and aspects appear. The evolution of the ontology and its updates imply updates of the bases and inferences that were built upon it. Past is a heavy burden, for instance let us consider the distribution of the annotations: a change in the ontology may change the shared space on which the distance is based and will modify its results and thus what should be done with the existing distribution?

First and without considering the consequences of the ontology changes over the other objects, the evolution of the ontology object itself conceals interesting work to be done. Before an evolution must be done, the need for it must have been detected. I identified three ways of detecting such a need:

- *feedback from usage*: instead of restraining the user to the controlled vocabulary of the ontology, one could imagine an option "new other concept" in the interface that would allow users to create and use a concept they believe is missing. The proposed new concept would be placed in the proposed location in the taxonomy, with a compulsory definition attached to it, most importantly, it would be marked as being a new temporary concept not yet validated (so that other users know it). Then, the event would be reported to the ontologist (recurrent report, mail notification, etc.), and would initiate a validation phase. Thus the ontology formalisation must include the primitives that will sustain its extension.
- *correlation analysis and clustering*: using clustering techniques [Delteil *et al.*, 2001b], one can analyse the patterns in annotations and facts described with the ontology. Recurrent patterns that do not correspond to a formally defined notion could be indicators of missing notions or of emerging interests in the organisation.
- *new terms and obsolescence*: maintaining an ontology (just like maintaining a memory) is not only about detecting new notions, it is also about detecting obsolete ones. To detect these aspects, one could envisage to use statistics over documents, retrieve keywords like in information retrieval to build index (removing stop-words); then, the differences between this index and the terminological level of the ontology could provide suggestions of improvements. The difference {index terms} - {ontology terms} ranked by chronological order of apparition and frequency could be an indicator of new missing notions. Conversely the difference {ontology terms} - {index terms} could be an

indicator of useless notions. These are only hypotheses, but it could be interesting to study their effectiveness.

Another source of maintenance will come from the references to other ontologies *i.e.,* inclusion or extensions of other ontologies in your own ontology (e.g. Dublin Core ontology); in the case of a change in a referenced ontology, repercussions have to be evaluated. This last point also raises the problem of ontology versioning, and mapping between versions. To allow transition between versions of an ontology, one could use a unification at the terminological level (with a TF*IDF approach for instance) to detect common stable parts and changed parts, assist the construction of update scripts and mappings.

Finally, the same structural modification may not trigger the same correction, depending on the associated semantics: the definitions and labels in natural language can be changed quite safely, the addition of a notion may require to review the facts using the subsuming notion to check if they could be specified, but this is not absolutely compulsory, the subsumption link modification may or may not need to mutate instances and check signature coherence, the signature changes may or may not require mutation of instances, etc. Thus patterns of transformations should be studied, classified to be able to provide building blocks for updates and coherence maintenance scripts.

13.1.1.3 Ontologies as interfaces, ontologies and interfaces

An ontology captures internal conceptualisation of a knowledge-based system and its links with the external conceptualisations of human users. Ontologies provide the semantic grounding for communication and as such, provide a conceptual interface between the conceptualisation of the system and the one of the user, thus they need to be understandable both to humans and to machines; otherwise they can no longer play the role of an interface and they are no longer usable. The ontology must not be restricted to its formalised form and a means for symbolic manipulation to mine and prepare spaces of wild information. It should not be thrown to the face of the user not only because its logical face is abstruse, but also because as humans, we do not mobilise and are not aware of our own whole conceptualisation at a given time: we focus. I could not picture my own conceptualisation of the world even if I wanted to: we focus and interfaces should focus with us.

This means a number of problems have to be tackled:

- *customising and filtering the access to the ontology*: I have made some experiences with the user profile-based filtering of the ontology, but much work is needed here to learn preferred terms and concepts of a user.
- *tutoring new usage*: learning and teaching the use of tools based on ontologies, the abilities they offer, the discovery of the underlying conceptualisation and the tasks they support, the ways to provide feedback and participate in the life of the ontology, and the interest to do so.
- *the visible tip of ontology*: representation and interfaces between the ontology and the user, going further than the fish eye / hyperbolic trees, acknowledging that the system usage should not be model centric, but model-supported.

The issues of the exploitation of ontologies in interfaces underlines the importance of semiotics consideration in ontology engineering. Interfaces have the unenviable role of bridging the gap between explicit conceptualisations captured in ontologies and day-to-day use of signs to denote concepts with unavoidable ambiguity and fuzziness. Users do not mobilise the whole conceptualisation each time they communicate, think, act, etc. therefore, a system must not impose to users to handle the whole ontology each time they have an interaction.

The ontology is at the interface between a symbolic system in its virtual world and cognitive agents in their real world. Different types of interactions and users imply different forms of access and different views of the ontology. The very simple fact of choosing labels in an ontology introduces the ontology in the field of interfaces. Thus interface and ontological problems must be tackled in parallel. This brings back the problem of choosing surrogates for representation (as in the field of information retrieval) and the concept of unique key (as in the field of databases). These aspects are mostly overlooked while, in my opinion, they are vital to allow the interpretation associated to a

model. Here is, for instance, a possible choice of key properties to enable visual selection of some notions through a surrogate:

Notion	Key surrogate properties	Other properties
employee	first name, family name, department	hiring date, office, activities, ...
person	first name, family name, NISN	age, height, address, ...
document	title, authors, date	format, ...
book	{*idem* document} + editor, ISBN	number of pages, ...
web page	title, URL	date, size, ...
department	designation, location	activity, members, ...

Table 36 Example of key attributes for notion surrogates.

The introduction of surrogate representative properties in the ontology and the study of their mechanism of inheritance would allow systems to generate surrogates for any annotated resource and present result in an adequate form. The general process could be:

- for every concept of a submitted query the system expands the query with optional questions requesting the properties participating to the surrogate of this concept e.g.: if a query contains a concept *employee* the system adds optional requests for the *first name*, *family name* and department of this *employee*, so that, if these information are available, they will be included in the result.
- for every concept type, the system has a presentation template (e.g. XSLT template) that encodes how the concepts should be represented using the properties participating to its surrogate. These templates are recursively applied to the structure of the result to dynamically build the corresponding compound surrogates. Labels of properties and concept types as well as natural language definitions contained in the ontology would also be used to generate these surrogates.

But the notion of surrogate representative properties goes beyond the graphical interface concerns and meets the notion of identity condition as defined by [Guarino and Welty, 2000] (see 2.2.5.3 page 88). In fact the notion of unique identifier (e.g., URI, ISBN) is sometime very artificial and as shown by the previous examples of key attributes, we, as human, usually require several key attributes to ensure the identity of an object. Thus between different systems or different annotation points of view, the identity and merge of two instances may be detected by using these identity surrogates. This problem is, for instance, well know in the community trying to detect acquaintance networks: to detect if an author of a given paper is the same person than the author a given web page, the key attributes of the identity have to be chosen and sometime weighted and combined to see if they exceed a confidence threshold. [Guarino and Welty, 2000] identified this problem is about distinguishing a specific instance of a certain class from other instances by means of a characteristic property, which is unique for it (that whole instance). They said it relies on a relation ρ satisfying $\phi(x) \wedge \phi(y) \rightarrow (\rho(x,y) \leftrightarrow x = y)$. Rather than a single characteristic property, an identity condition can be based on a group of properties and complex test conditions, to establish the identity. As we saw in 3.4.2 page 130, the new formalisms for the semantic Web will include primitives to denote the equivalence of two instances, thus addressing the fact that the mere existence of a URI does not ensure that the pointed object as not been annotated through some other referencing system. Therefore equivalence rules could be envisioned to automate the detection of an equivalence. In Figure 147 are shown two simple examples of what such rules could be for "*book* instances". These rules could be applied on the bases or taken into account by the an extended join operator that could thus join annotations on the basis of such extended joint criteria. Finally, these rules are scripts for the equivalence of primary keys but they express identity conditions which are pieces of knowledge independent of any instance and defined at the intensional (class) level; thus this is ontological knowledge that should be included in the ontology and taken into account along its whole life-cycle.

```
IF
 [Book: ?b1]->(Title)->?t1
            ->(Author)->?a1
            ->(EditionNumber)->?e1
AND
 [Book: ?b2]->(Title)->?t2
            ->(Author)->?a2
            ->(EditionNumber)->?e2
AND
 Trim(UpperCase(?t1))=Trim(UpperCase(?t2))
AND
 Trim(UpperCase(?a1))=Trim(UpperCase(?a2))
AND
 ?e1=?e2
AND
 ?b1≠?b2
THEN
 Equivalent(?b1,?b2)
```

```
IF
 [Book: ?b1]->(ISBN)->?i1
AND
 [Book: ?b2]->(ISBN)->?i2
AND
  ?i1=?i2
AND
  ?b1≠?b2
THEN
  Equivalent(?b1,?b2)
```

Figure 147 Examples of rules for identity equivalence

Finally, as paradoxical as it may seem ambiguity is vital to access the ontology. The ontology is a neatly formalised theory, but the ways of access to the ontology must take into account fuzziness and ambiguity and capture them so as to exploit them in interactions with the user. Ambiguity and changes should be modelled and captured just like any domain, to be exploited in interfaces with the user. When we consider that ontologies are used to disambiguate, it does not mean that they are used to eradicate ambiguity in communication (this is not possible with humans and natural language). For me, it means that ontologies must understand and capture the ambiguity in an explicit form, and that we have to:

- understand where ambiguities are, their nature, their sources and their mechanisms,
- incorporate and include this understanding in the representation of ontologies,
- design mechanisms that exploit this explicit understanding to disambiguate communications.

The ontology must include the primitives that will support the interfaces between it and the real world.

13.1.1.4 Conclusion

The ontology must include the primitives that will support the different stages of its lifecycle. This may imply to differentiate facets of the ontology and study their relations (exploitation facet, maintenance facet, design rationale facet, representation facet, etc.). It is important to acknowledge the existence of multiple points of view both in the conceptualisation and in its use; for instance, I believe that the difference between what it is useful to capture and formalise for ontology design and what it is useful to capture and formalise for ontology exploitation is important and should be dealt with.

Finally, ontology engineering will never make it to the real world if we do not teach on the subject; history showed that in the past artificial intelligence solutions suffered from the lack of trained people to apply and maintain them; there is no reason for the ontology to escape that very same trap if we do not address it.

13.1.2 Ontologies for MAS *vs.* MAS for ontologies

The first section concentrated on the ontology lifecycle generic problems. But as I asserted several times, multi-agent systems and ontologies can mutually benefit to one another.

13.1.2.1 Use of ontologies for multi-agent systems

We have seen all along the work in CoMMA that ontologies play a pivotal role in the definition of communication languages and protocols. I believe we should go further and envisage to try to build ontologies of the multi-agent domain. This could provide a semantic useful grounding:
- in *agent management*: to build agent management systems, actions allowed on the agents, representation of agents, etc.
- in *service matchmaking*: to describe services to be advertised, build inference for matchmaking, structure yellow pages, etc. LARKS [Sycara *et al.*, 1999b] is an example of language trying to tackle agent services description and matchmaking.
- in *agent security*: types of agents, description and identification attributes could be captured and used for security enforcement for instance in mobile agents systems.
- in starting up an interaction: to recognise the protocol and message format, extensible ontologies of speech acts, etc.
- in *rationalising methodologies and unifying notations*: this could allow the design of generic tools, of libraries of patterns (types of societies, structural patterns and types of relationships, roles and their characteristics, protocols, etc.)

An ontology of multi-agent systems also appears to me as the natural first step to the introduction of some reflexivity and introspection in the system.

13.1.2.2 Use of multi-agent systems for ontologies

Ontologies can benefit from multi-agent system too. The first application that comes to mind is the implementation of the Computer Supported Collaborative Work platform to build and maintain ontologies. A distributed multi-agent platform could be envisaged:
- to manage the negotiations over the ontology through specific protocols.
- to propose wrapper and mining agents in charge of importing terms from heterogeneous sources (mining the internal web, mining database schemas, wrapping online dictionaries etc.).
- to manage the propagation of the changes in the ontology.

etc.

In the ontology use phase also, we could use multi-agent systems to support use, monitoring and maintenance of the ontology:
- to collect feedback from different uses.
- to look for emergent patterns and correlation in repositories,
- to allow multiple ontologies and build gateways between them

etc.

13.2 Organisational memories and multi-agent systems

As shown in Figure 2 page 33, a memory has a lifecycle including the one of the ontology. Tools are needed for every stage. I have presented agents that participate in the construction, diffusion and evolution, allowing users to: annotate resources, and diffuse them through push and pull technology.

In my opinion, agents can be used for the following phases:

- *construction*: the heterogeneity of sources calls for the development of *wrapper agents* (wrapping legacy software or resources) and especially *miner agents* (wrappers mining semi-structured or unstructured sources), *adapter agents* (adapting other information sources: database agents, lotus note agents, librarian search engine, search engine, etc.)
- *diffusion*: we meet here the discussion on interfaces of the previous section and the previous chapter. Moreover, I believe that a study of interfaces for presenting information with *context awareness* is necessary. Multiplying diffusion modes, the information content can be presented in very different ways depending on the content, the importance, etc. For instance a screen saver can be used to push the news bulletin, or social announcements (non intrusive way of presenting information), life-like characters can be used to give guidance on a task or report an error (ephemeral task related information), daily digest of new resources found by the system can be sent by e-mail in the morning before arrival (non intrusive push way, but leaving a trace), etc.
- *capitalisation*: a good diffusion favours capitalisation. Additionally co-operation can be fostered by agents in implicit ways (collaborative filtering, automation of word-of-mouth, etc.) or more explicitly using push technology to suggest resources to be consulted, profile of other members of the organisation that have equivalent activities, interest, etc., proposing expert matching services to solve thematic problems, setting up similarity-based searching services like in case-based reasoning, etc. Additionally, workflows, practices and information flows can be captured and used to drive proactive suggestions of the system and accelerate diffusion and use of knowledge over the organisational structure. For instance a new automatically detected resource (wrapper) could be systematically reported to an expert of the domain (matching profile and extracted information) that would validate and complete the annotation before it is pushed to all the employees having a corresponding interest in their profile.
- *evaluation*: I already suggested to solicit feedback from the user, and in CoMMA such feedback is necessary to the machine learning algorithm. However feedback can also be used to build monitoring reports sent to the administrators or to the persons in charge of knowledge management (knowledge chief officers, technology surveyors, chiefs, etc). Other indicators could be monitored such as failed queries, recurrent search and annotation patterns, usage statistics etc.
- *evolution*: in addition to the construction tools, evolution requires modification and forgetting capabilities to tackle obsolescence. I already suggested some ways to implement propagation of updates and maintenance of coherence; interfaces are also needed here. Reports on annotations and sources barely used, or detected needs will trigger removals of some annotations or wrappers and addition of new ones.

In the following subsections, I focus on some particular ideas I am interested in.

13.2.1 Opening the system: no actor is an island

13.2.1.1 Extranets: coupling memories

No organisation is an island and its activity leads it to collaborate with other organisations. This configuration is now called the *virtual enterprise* and when it happens the collaborating organisations may want to share some of their information to facilitate exchanges.

Coupling organisational memories imply to set-up gateway agents (see Figure 148) creating a junction to:
- ensure security and contextual filtering based on the nature and scope of the collaboration,
- deal with semantic frictions: different formalisms, ontologies, etc.

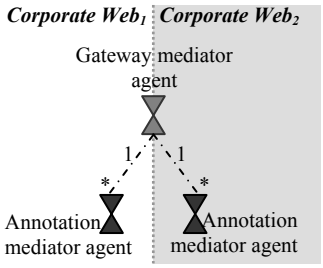


Figure 148 Gateway mediator agent

We are considering the problem of mapping two ontologies. The tests we currently carry out are based on an adaptation of TF*IDF to the problem of ontology reconciliation. I called it TF*INF for "term frequency inverse notion frequency": it uses a term space to represent a notion by a vector of terms used in its labels and definition; then, vectors of the two different ontologies are compared and the closest ones are kept using a threshold on the cosine measure (cosine>0,55 in our tests). From this mapping initialisation, we envisage iterative refinements (structural constraints, feedback of knowledge engineer) until the mapping is acceptable. Then, translation scripts should be produced to allow translation of queries and answers from one corporate web to the other and vice-versa.

The ontological overlap defines the possible domain of interactions; but sometimes conflicts arise.

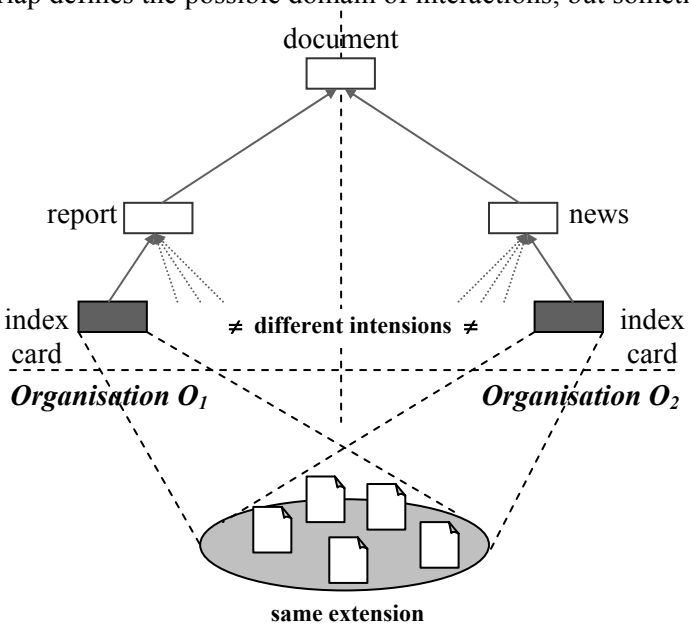


Figure 149 Conflict between two ontologies

In CoMMA, a problem of incompatible points of view appeared, as illustrated in Figure 149. Two telecom companies disagreed on the definition (intension) of an index card *i.e.,* a document used for reporting a technology monitoring event. However they agreed on the actual instances of the document (extension). It means that they agreed on the extension of the "index card" concept, even thought they disagreed on its intension.

In this case, a simple merge or consensus is impossible; try to reconcile would be an error since both organisations agreed only on one point: to say they disagree. However the extensions being the same, they may want to be able to exchange knowledge about these index cards. In that case, I can see two options:

- option 1 *generalise*: each time we exchange knowledge about index cards we generalise the type to a document and exchange knowledge about documents. But this is a pity since we know the extensions are the same and we loose the typing information which may result in less precision and recall.
- option 2 *equivalence of extension*: we create a rule saying that the extension of *index card* of the ontology of organisation 1 in its memory, is equivalent to the extension of *index card* of the ontology of organisation 2 in its memory ($\texttt{index\_card}_{O1}\texttt{(x)} \Leftrightarrow \texttt{index\_card}_{O2}\texttt{(x)}$). However, their intensions are not the same ($\texttt{index\_card}_{O1} \neq \texttt{index\_card}_{O2}$). Thus intensional knowledge such as relation signature must are either generalisable to the closest upper concept equivalent in both ontologies (here *report* or *news* if they have an equivalent notion in the ontology of the other organisation, or *document* otherwise) or lost during exchanges between the two organisations (*i.e.,* it is specific to the intension of *index card*).

This second option is close to multi-instantiation since a second facet of an object is generated, using another intension; I call it *instance migration* enabling actors to exchange instances at a semantic friction point by using template for translation of shared characteristics such as the title, author, etc. This migration could indeed be implemented by maintaining multiple instantiation of the shared objects, from both organisations' point of view (*i.e.,* their ontologies).

13.2.1.2 Web-mining: wrapping external sources

The effort of annotation in CoMMA was part of the role of some stakeholders (technology monitors, mentors). No organisation is an island, it is included in a culture, a country, a society, a market, etc. and lot of interesting information will be available on the open Web about the organisation's environment, core activities domain, etc. Thus I envisage here a way to provide semi-automatic assistance to generate annotations from external sources. I see it as the dual problem of the usual vision of corporate portal. A corporate portal usually offers services from the organisation to the outside-web, it is the shop window of the organisation; I call it the *outer portal to the inner services*. Conversely the organisation resources can be used to filter and mine the outside world *wild* web and provide its internal communities of interest with a portal enabling them to access carefully chosen, selected and validated external sources providing resources interesting for them; I call it the *inner portal to the outside services*. A way to implement an inner portal is the use of wrappers as shown in Figure 150; the wrappers I shall consider here are Web page miners.

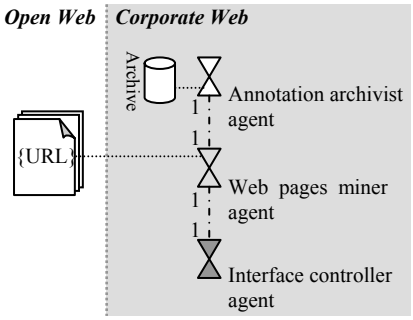


Figure 150 Web pages miner agent to implement inner portal.

A Web page miner applies content-based filtering and data-mining techniques, to develop the other side of the portal *i.e.,* from inside to outside, using the knowledge and competencies of the organisation to filter and harvest the outside world.

The open Web is human-intended and full of unstructured or semi-structured resources. The extraction of semantic annotations used to provide the organisation with internal pointers to the outside relevant resources raises the problem of specific wrapper development for each source. This led researchers to consider generic wrappers and parameterisation scripts or customisation workshop tools where one can quickly develop a new specialised wrapper using automatic rule generalisation and other machine learning and text mining techniques.

The technology monitoring scenario in CoMMA led us to consider such a special type of agent deriving from the wrapper agent that extracts content from unstructured external sources and performs appropriate data conversion. They would be hybrid between wrapper agents and data analysis agents. We identified two options:
- On-the-fly conversion where the wrapper agent uses its skills to convert information whenever it is solicited. This approach has the advantage of always providing the requester with up-to-date information, but the conversion process may slow the agent's answer.
- An image generator wrapper, triggering checks at chosen intervals and, if needed (e.g. a new version of the monitored Web page is detected), applying conversion mechanisms to update a local structured image of this data inside the corporate memory. This approach has the advantage of providing information very quickly since the agents work on the structured image of the information and to be available even if the outside Web site is off-line. However, depending on the parameter settings, the data may not be up-to-date (e.g. cyclic update every 30 minutes) and also the amount of information duplicated in the intranet may be important.

These agents would be a special kind of Annotation Archivists just systematically refusing to archive other annotations than the ones they generate through Web page mining.

My preference would go to the second option for two reasons:
- It is fast when dealing with a query since the structured data are always directly available when needed.
- It decouples and isolates the intranet from the Internet, which is an appreciable feature.

In [Bergamaschi and Beneventano, 1999], a mechanism is described for generators of translator: "the agent is generated based on the description of the conversion that needs to take place for queries received and results returned." The idea of facilitating the annotation generator design is indeed interesting since agents have to be customised for each new source. A library of useful functions or a toolkit for Web Wrappers could be a valuable asset for feasibility and acceleration of the process of developing such agents. For example, [Muslea *et al.*, 1999] describes an approach to wrapper induction "based on the idea of hierarchical information extraction" where extraction rules are described as finite automata learned by the agent. This would allow users to develop and launch a population of agents, each monitoring an assigned source.

We are currently implementing and testing a first prototype with the following approach:
- in the Interface Controller, the users select a source of web pages having a similar semi-structure;
- the users annotate one of the pages, giving an example of RDF annotation;
- the Interface Controller derives an XSLT template to extract information from the Web page and automatically build the corresponding RDF annotation;
- once the template is validated, the Interface Controller requires the creation of a Wrapper and gives it the template and the sources it should be applied to;
- the Wrapper creates its base, registers with an Annotation Mediator like any Annotation Archivist and is ready to participate to query solving;
- the Wrapper maintains its base, monitoring changes in the source.

The approach supposes a rather stable format of annotation for a set of pages. Further improvements to further generalise the template could rely on machine learning techniques.

A remark must be made, though. The quality of the information retrieval process is critically dependent from the quality of the annotations: while CoMMA uses inference libraries to exploit the ontology in widening and narrowing users' queries, it does it relying only on the RDF annotations, without even considering the actual document text. Relying on automatically generated annotations may not provide the required quality of the semantic annotations and a semi-automated environment in which a human operator is still involved, as proposed here, is the warranty of an acceptable quality.

13.2.1.3 Collaborative filtering: exploiting the organisational identity

No organisation member is an island. Like-minded people choices may be used to suggest consultations to a user: if we share some interests, then my consultations, my finding in information retrieval may be interesting for you and vice-versa. Collaborative filtering for recommending was not used in CoMMA, but was studied in other projects (see state of the art) and should be used in a complete solution to improve push mode and pro-active behaviour of the system, that fosters the capitalisation of knowledge. Several approaches seem interesting to me:

- *Memory event notification*: the monitoring of memory events to keep user informed of new resources available: In CoMMA, we are interested in introducing 'proactiveness', therefore, we identified several possible roles. We said the UPM is in charge of using and updating the profile when the user is logged on. It should perform learning on the fly, but also proactive actions (e.g. register an aborted query to be reconsidered later). As soon as the user logs off, another role called User Profile Processor is in charge of working on the user's profile to analyse it and try to proactively identify interesting and relevant information (e.g.: launch complex queries derived from the user's queries and profile). These first ideas are at user's level.
- *Like-minded people suggestion*: collaborative information filtering is defined in [Klusch, 1999c] as the automation of the process of 'word of mouth' in a community to anticipate the individual needs and uses relying on past uses of other users. Some collaborative filtering techniques are described, for instance in [Guttman *et al.*, 1999]. The main idea is the comparison of user profiles, consultation traces, and feedback to detect similarity patterns and suggest new actions (in our case a consultation) to users.
- *Communities of interest* (CoIn): we could also envisage an explicit clustering of profiles and a detection of acquaintances to enable users to visualise the communities or the groups of interest, monitor them, register to one of them (as in newsgroups or mailing-lists) and get acquainted with like-minded people. These groups could be managed by agents in charge of running recurrent queries concerning the interests of the group and keeping up-to-date a document containing the resulting survey.
- *Expert matching*: finally, by relying on user profiles, communities of interests and acquaintance networks, one could envisage to propose expert-matching services where an agent proposes a list of competent people that could be consulted for a problem described by given topics.

13.2.2 Closing the system: security and system management

While the previous part emphasised the need to open the system and to foster collaborations, a given number of security and administration problems would have to be tackled in a real-case solution:

- *Management of user profiles*: the rights to add a new user, modify a profile, visualise a profile, use it in query solving, etc. are not allocated to the same role; the addition of a profile requires administration rights while the information derived from usage are highly personal and should be visible and modifiable only by the user.
- *Deployment and configuration*: the deployment and maintenance of a configuration of a multi-agent system in an organisation will require administration tools and security. One of the perspectives that appeared in CoMMA is that we could augment the content and use of the corporate model to drive the deployment of a configuration, provide pictures of the mapping between the structure of the human organisation and the agent organisation. This augmented model would really anchor the system in the organisation and augment its awareness of the organisational environment and infrastructure. Indeed, the use of corporate structural model could improve co-operation mechanisms e.g. an Interface Controller could require an Annotation Mediator in a finance department because the query it has to submit is more likely to be in its area; this negotiation could rely on a contract net protocol [Davis and Smith, 1983] and an ABIS for the Annotation Mediator obtained by making the union of the ABISs of its Annotation Archivists.
- *Management of archive modification rights*: addition, edition, deletion of annotations, will require the management of rights over the archives with, most probably, delegation to local administrators. Additional administration tools for managing the beneficial oblivion in organisational memories and take into account obsolescence of knowledge are also needed.
- *Management of archive consultation rights*: finally, the access to annotation base is actually controlled by the federation of Directory Facilitators and Annotation Mediators. In a human organisation it is necessary to be able to specify the visibility range of an information *i.e.,* to control who can see and query an annotation, and who cannot. Without this ability, critical information may never be saved in the memory, for fear that "a colleague working in the same domain will use it before and get promoted ☺". Read-access rights for groups and others under UNIX-like network operating systems illustrate this needed functionality.

For these management aspects, an ontology of rights, events, roles and actions in the organisational memory could be very useful. It would provide the vocabulary to describe the security policy.

13.2.3 Distributed rule bases and rule engines

While being extremely useful, factorised knowledge raises problem for distributed bases and distributed query solving. Let us consider the following example where there exists a definition of *son*:

$$son(x,y) \Leftrightarrow person(y) \wedge person(x) \wedge male(x) \wedge child(x,y))$$

and where some facts are distributed, for instance:

site (1): $person(x) \wedge person(y) \wedge child(x,y)$

site (2): $male(x)$

Then, the rule will not apply on any of the sites, thus we need some kind of distributed protocol of inference engine to apply axiomatisation. At least two approaches can be chosen:

- *augment bases*: Annotation Archivists perform local application of rules and local augmentation when possible, and Annotation Mediators try to detect distributed cases and generate corresponding complement to annotations that are then archived by Annotation Archivists.

- *augment queries*: we could choose to not modify the base, but to extend queries and submitted annotations to a normal form where all defined notions have been replaced by their definition. thus rules would only be applied at the interface of the annotation-dedicated society and the other societies *i.e.,* the Annotation Mediator would inflate incoming annotations and queries and deflate out-going results.

The decision requires a review of the work done in distributed reasoning, but also to consider other problems such as non monotonous reasoning, temporal reasoning, etc. if such expressiveness was to be needed.

The introduction of rule bases also creates the problem of the management and distribution of these rules in a distributed system. While this remains an open issue in CoMMA, the role of rule base management seems equivalent to the ontology agent since it offers the other agents the ability to download rules and to update them as needed: it is in fact, the axiom part of the ontology.

Finally, I suspect that different organisations and sub organisations will have different uses for rules and inferences. Thus it could be interesting to study the concept of communities of formalisation and communities of inferences in which different aspects of the formalisation are activated while others are discarded as useless in this community of users.

13.2.4 Workflows and information flows

Organisations have policies and views that allowed us to suggest that an ontological commitment was possible. Likewise these policies find expression in workflows and information flows in the organisation. These processes are different from one organisation to another, participate in knowledge capitalisation and govern the co-operations between the employees. To acknowledge and assist this aspect, it would be interesting to envisage, for instance, a *generic workflow agent* that could be customised with scripts to foster such processes. I found an example in the APROBATIOM project I was involved in, where documents followed some processes with stages and roles attached to each subtask, and the system we were designing had to assist the progress of this process for each instance of document, project, etc. Thus we could envisage monitoring agents, in charge of detecting documents in a certain state and pushing them to the next person that can continue the process.

13.2.5 New noises

Finally, I did explain how ontology and knowledge-level reasoning could improve recall and precision respectively reducing silence (*i.e.,* some relevant answers are missing in the results) and mismatch noises (*i.e.,* some irrelevant answers are present in the results). However, the use of semantic search engine revealed new categories of noises.

The first category is known as the *lack of common knowledge*. It is already addressed in knowledge modelling using axiomatisation. The symptoms are the silence of the search engine due to a missing inference of equivalence, and the remedy is the addition of formal definitions such as:

$$son(x,y) \Leftrightarrow person(y) \wedge person(x) \wedge male(x) \wedge child\ (x,y))$$

A second noise experimented in CoMMA is the *over constrained queries* research mechanisms. Some information required in a query are vital constraints (e.g. the name of the person) while others are "nice to have if they are available" (e.g. the e-mail address, mobile number). This leads us to differentiate between constraining questions and "nice to have" questions using `$option` operator that indicates an optional part of a query. In Figure 151, the query requires persons interested in computer science, with their name and, if it is available, their first name.

```
1   <CoMMA:Person>
2    <CoMMA:FamilyName>?name</CoMMA:FamilyName>
3    <CoMMA:FirstName>?firstname $option</CoMMA:FirstName>
4    <CoMMA:IsInterestedBy> <CoMMA:ComputerScienceTopic/> </CoMMA:IsInterestedBy>
5   </CoMMA:Person>
```

Figure 151 Query with an option part

A third noise experimented in CoMMA is the *lack of conciseness* in result generation. For instance, in response to the previous query, the system may answer something like "Fabien Gandon is interested in Multi-agent systems", "Fabien Gandon is interested in Java", "Fabien Gandon is interested in Web Technologies", etc. We, as humans, would build a more concise answer regrouping at least the characteristics around the focus of the question, which, here is the person. So we would at least be more concise by saying "Fabien Gandon is interested in Multi-agent systems, Java and Web technologies". For that purpose the operator `$group` has been introduced to point the focus in the query and group the results according to the instances of the given concept. By default, a heuristics places this operator on the root of the query.

Finally, there is the *noise of evidence*, which is very hard to chase. This noise corresponds to results presented by the system that are evident to the user or the uselessness of which is evident. Trivial results, answers already known re-demonstrated or re-found are symptoms of a lack of knowledge about the difference between what should be shown and what is already known. As a human, if I suppose my interlocutor knows and is familiar with a certain subject, I shall make my answers concise. Otherwise it is a noise in communication which, in the best cases, is considered as an exasperating habit of making interminable speeches or, in the worse cases, is considered as an offence whereby I implicitly suppose the other's lack of knowledge. The only way to avoid that, is to (a) factorise answers by using axioms of common knowledge, short time memory mechanisms and focus in order to avoid repetitions and (b) profile-based filtering to remove or hide obvious knowledge according to interlocutor's expertise in an area. The second type of evidence noise, where the uselessness is evident appears with inferences that derive logical results perfectly true from a symbolic manipulation point of view, but perfectly useless for the user. Consider the following rule:

$$\text{colleague}(x,y) \Leftrightarrow \text{person}(y) \wedge \text{person}(x) \wedge (\exists\, o\ \text{organisation}(o) \wedge (\text{include}(o,x) \wedge \text{include}(o,y)))$$

Left as it is, it will generate results such as `colleague(Fabien, Fabien)` because it is not precise enough (the $x \neq y$ is missing) and thus axiomatisation calls for more axiomatisation to avoid noises. It will also generate more *"colleague"* than wanted since the *"include"* relation is transitive and thus if the INRIA is in the W3C, consortium I end-up being the colleague of Tim Berners-Lee while this was not the *intention* nor the *intension* of the *"colleague"* relationship. This requires to differentiate direct inclusion from indirect inclusion.

Likewise the algebraic properties of relations raise the same problems. For instance, if `SeeAlso( , )` is declared symmetric and transitive, then when calculating the transitive closure, the system derives that the relation is reflexive: `SeeAlso("Dune", "Dune")` while we do not want it.

$$\text{SeeAlso}(x,y) \Rightarrow \text{SeeAlso}(y,z)$$

(e.g. SeeAlso("Dune", "Herbert") ⇒ SeeAlso("Herbert","Dune"))

$$\text{SeeAlso}(x,y) \wedge \text{SeeAlso}(y,z) \Rightarrow \text{SeeAlso}(x,z)$$

(e.g. SeeAlso("Dune", "Herbert") ∧ SeeAlso("Herbert","Dune") ⇒ SeeAlso("Dune","Dune"))

A way to avoid this is to declare the relation as an *irreflexive* relation *i.e.,* `SeeAlso(x,x)` is an error if encountered in an annotation as a genuine fact or is discarded if obtained as the result of an inference. It is equivalent to replace transitivity by *strict transitivity*:

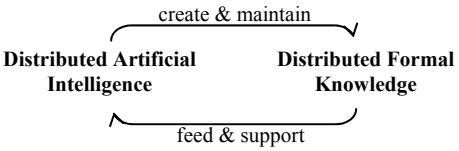
$$\texttt{SeeAlso(x,y)} \wedge \texttt{SeeAlso(y,z)} \wedge \mathbf{x \neq z} \Rightarrow \texttt{SeeAlso(x,z)}$$

We can also notice here that the algebraic properties of relations raise a lot of discussion concerning their ability to be inherited.

13.3 Dreams

I have always been amazed at the way an ordinary observer
lends so much more credence and attaches so much more
importance to waking events than to those occurring in dreams...
Man... is above all the plaything of his memory.
— André Breton

I have a dream that one day a nation of agents will rise up... to manage knowledge distributed over the 'world *wild* webs' ☺. More seriously, I believe in the *absolute complementarity between distributed artificial intelligence and distributed formal knowledge*.



Natural complexity of reality and above all artificial complexity of the conceptualisation developed to account for this reality led mankind to automation and computerised real artefacts to achieve tasks in this real world. Then, through data storage, networks, information systems and online services, we created new worlds that are rapidly growing out of any centralised control. This complex virtuality leads us again to automation and computerised virtual artefacts to achieve tasks in this virtual world. Rising up through abstraction, we are developing new paradigms among which agents are a candidate for a new level of abstraction that appears appropriate for the new family of distributed complex problems raised by the growing wild information landscapes.

To those who assert that nothing that is done with agents could not be done with objects, I say that is logically true; but then, what can be done in objects that cannot be done in a good old procedural language, etc. ? at the end of the day, everything goes down to assembly and then machine language...

So the question is not "is the implementation feasible ?", the question is rather "how easy can the implementation be ?". The reason for rising up in abstraction is to reduce the gap between on the one hand the conceptualisation of a problem and of its solution and on the other hand their formalisation in an implementation language. By considering autonomous entities, complex organisations and protocols as first-class natural citizens of the implementation language, agent technologies can offer an additional layer of abstraction. The main goal of artificial intelligence is to capture intelligent behaviours (from simple tasks to complex ones) in artificial systems. Multi-agent systems try to do the same at collective level to produce artificial societies. This is not a plea for blind agentisation of everything. In my opinion, the future is hybrid, many times in that Ph.D., we saw that the solution was on a continuum between two extremes, all types of agents are needed because all types of tasks are to be done and all types of resources are to be managed; only variety can tackle variety. And in fact, *a real implementation of the distributed artificial paradigm should encompass from the most simple types of object, data and interaction to the most complex types of artificial intelligence entity, formal knowledge and social rules*. (see Figure 152). Moreover, the agent paradigm should not only consider the integration of established paradigms like the object-oriented design or the component-oriented design as it has been done in some frameworks, it should also look at other emerging paradigms like pattern-oriented design and aspect-oriented design that may bring solutions to some difficulties encountered by the current implementations of agent frameworks.

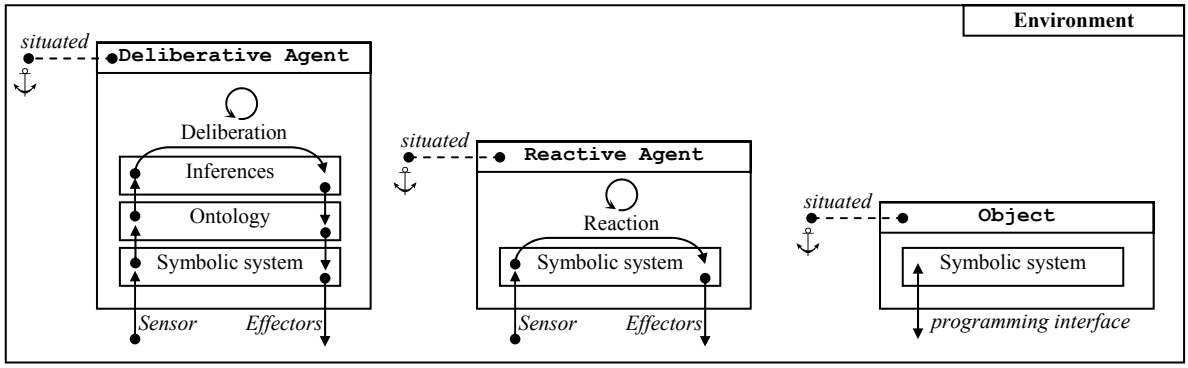


Figure 152 The three atomic entities of a complete distributed artificial paradigm

We saw that formalisation of knowledge depends on the inferential needs and on the adopted conceptualisation. Thus multiple forms of formal knowledge will continue to exist. A component-based approach to the expressiveness of formal languages and the design of associated inference engines seems to be the right path to take. User will choose the expressiveness and pay the associated calculation price to match their needs.

Interconnection will require semantic alignments of the underlying ontologies and time will require the management of ontology evolution. I believe this will call for reconciliation of on the one hand the constructivist facet of ontologies and knowledge and on the other hand the cognitivist facet of ontologies and knowledge. In the cognitivist facet lies the dependency of symbolic distributed artificial intelligence on ontologies: they can provide powerful explicit conceptual models to support automatic systems in operating meaningful manipulations and communications. Conversely, in the cognitivist facet lies the dependency of distributed knowledge and consensual ontologies on distributed artificial intelligence: they can provide powerful lifecycle supporting systems while maintaining what has been built upon them. A system that buys the ontology approach will have to pay the price for it and given that symbolic distributed artificial intelligence and distributed formal knowledge are indivisible, it seems there is a need for a unified paradigm and associated complete frameworks.

Even if I see a lot of problems that remain to solved, I believe agents and ontologies are here to stay. Will these paradigms be called 'autonomous agents' or 'holons' or 'artificial entities', etc., 'formal ontologies' or 'logical conceptualisations' or 'semantic theories', etc. it does not matter to me. The terms may change, but I believe the concepts will remain.

The way these concepts will spread could be the one followed by the internet technology in the 70s and 80s also followed by the World Wide Web in the 90s *i.e.,* start with small corporate semantic web and agent systems, connect them to build semantic extrawebs and slowly grow to world-wide semantic web and multi-agent systems. The semantic webs are a favourite virtual world for information agents to be deployed and web services are a favourite application domain for agents to be used.

A collective memories is a *distributed* system in the broad sense of the term: it is not only distributed between computer systems; its distribution includes other artefacts as well as persons. Solutions for knowledge management must acknowledge this *super distribution* to integrate our reality in a symbiotic way. The model of distribution must go beyond the technical dimension and account for the real world they interact with: the cultures, the organisations, the infrastructures, the environments, the people, etc.

In parallel to this growing diffusion of the concept, I would love to see a diffusion in depth: lower and lower to the heart of the computer systems. More and more software are using metadata: mail headers are growing fast with additional metadata, office tools save our documents with a lot of metadata, web-pages of course include more and more meta-data, etc. I believe that the good old fashion tree structuring of storage is going to look pretty static and poor compared to the search abilities provided by the metadata; some users are already using operating systems search capabilities more than file managers folder views. One of my long-term dreams is knowledge-based operating systems where structure of mass storage is flat, but indexing is rich, integrating all the resources through semantic annotations in unified customisable schema, whichever application generated them. Knowledge would smoothly come and leave the system, for instance e-mails and attachments would be annotated and stored like any other file and would travel with rich metadata. Multiple views could be generated at will to present a structure browse and search the user's personal storage in the way the most adapted to the task the user is performing.

When one claims to take people off into the imaginary, one has to be able to bring them back to reality... and without damage.
— Raymond Devos

Conclusion

What we call the beginning is often the end.
And to make an end is to make a beginning.
The end is where we start from.
— T. S. Eliot

Concluding a Ph.D is to conclude the theses that were defended and open the perspectives with theses to be defended. Therefore, let us review the theses of this work.

Semantic webs can provide distributed knowledge spaces for knowledge management. Classical keyword-based search engines are limited to lexical operations on the terms denoting instances of the use notions. The introduction of semantic annotations and models may guide systems in exploiting information landscapes and allow them to simulate intelligent behaviours improving their performances. I have shown how both aspects of the knowledge management issues can benefit from a semantic web framework to implement these improvements:

- the *aspect of a persistent memory*: a semantic annotation is a knowledge-level indexing structure that allows systems to use several indexing sources (manual, wrapping, mining, etc.) and to manage heterogeneous indexed resources through their URL (corporate web documents, reference documents on the web, etc.) or URI (people, organisations, physical resources such as books, etc.). Annotations play both central roles of persistent repository of acquired knowledge (the one that is formalised in the annotation) and persistent indexing of heterogeneous information resources; together annotations and resources constitute the persistent memory.
- the *aspect of a dynamic nervous system*: annotations are geared to distribution of information. The emission of an annotation anywhere in the organisation is an act of capturing knowledge. Their formal structure enables system to manipulate them and propagate them. When they annotate people and organisations, they can be used by the system to reason on its environment. In our case, an organisational model and some user profiles were described to guide the system in its global capitalisation of knowledge and allow the local adaptation and customisation to each user. Both types of models participates in the environment awareness needed by the system to match the application scenarios and manipulate annotations about information resources in an intelligent fashion.

> *By means of electricity, the world of matter has become a great nerve, vibrating thousands of miles in a breathless point of time ... The round globe is a vast ... brain, instinct with intelligence!*
>
> — Nathaniel Hawthorne (in 1851 !)

The CoMMA prototype was designed to apply this semantic web vision to the problem of a corporate memory. But I believe the approach and the techniques developed could inspire techniques and approach for more open systems. It could start with small corporate semantic webs connecting together to build semantic extrawebs and slowly grow to world-wide semantic web.

Ontologies are applicable and effective means for supporting distributed knowledge spaces. The experience I conducted in CoMMA showed both that it was feasible to build an ontology and what could be the process for doing it.

The whole process was driven by scenario analysis starting from informal heterogeneous resources and following a formalisation process described as an enriching process whereby initial informal lexicons are augmented with additional logical structures formalising the aspects relevant for the application scenarios. The different stages I experimented and detailed are: data collection and reuse; term analysis and generation of intermediary representations of the ontology; taxonomic structuring and ontology construction performed in an event-driven fashion against top-down, middle out and bottom-up perspectives; granularity improvement by defining rules factorising knowledge. Knowledge being holistic, its collection must be bounded: the coverage of the ontology is evaluated in terms of exhaustivity, specificity, and granularity against usage scenarios. Any missing coverage triggers additional collection and/or formalisation.

I developed an XML-based tool and used it to visualise different views and to browse the ontology during its construction. However I have stressed that ontologists definitively need a comprehensive integrated set of tools, to support the complete ontology lifecycle at each stage and for each transition. This is vital for the development and maintenance of a real-scale ontology-based system.

The resulting O'CoMMA is a prototype ontology used for trials and to initialise the building of new ontologies rather than a fully mature and polished ontology. However it already proved to be:

- *a component in itself of the corporate memory*: the ontology captures a knowledge about the company and the application domain. This knowledge is useful in itself and an explicit ontology provides a document presenting the organisation terms, policies and conceptualisation.
- *a vital support to inter-agent interactions and intra-agent operations*: in multi-agent systems, agents need to rely on a shared conceptual vocabulary to express and understand the messages they exchange. Based on this shared semantic space, complex interaction protocols can be specified. Moreover, the agent expertise in manipulating heterogeneous resources relies on internal architecture exploiting the semantic annotations and their ontology.
- *a vital support to inter-user interactions*: annotations and queries are based on the same shared ontology that provides a consensual conceptual vocabulary to expressed them. Just as it enables the agents interactions, the shared ontology also enables users' interactions allowing the query or the annotations described by a user to be compared with the annotations of another user, across the organisational structure.

I have detailed the content of O'CoMMA and explained how the structure of the ontology shows which parts can be reused, and what adaptation is needed depending on the changes in the scenarios and the application domain. I have also reported experiences of appropriation and partial reuse for other projects. The experience showed the usefulness of an existing ontology to initialise the construction of a customised ontology for a new application context and todemonstrate the additional power it brings to a solution. Finally, on several occasions, I pointed out that natural language processing tools are extremely useful to build or customise an ontology and, in particular, to populate application-specific parts such as the domain description.

Multi-agent systems are applicable and effective architectures for managing distributed knowledge spaces. The whole idea is the one of 'fighting fire with fire': in CoMMA, to the distribution of artificial knowledge, we opposed the distribution of artificial intelligence and to heterogeneity of (re)sources we opposed heterogeneity of agents.

I have detailed the design of such architecture for a closed system *i.e.,* a system where the type of agents is fixed at design time. It showed the effectiveness of a top-down functional analysis along the organisational dimension of the multi-agent system. It is an organisational approach in the sense that the architecture is tackled, as in a human society, in terms of roles and relationships. The architectural design I described, starts from the highest level of abstraction (*i.e.,* the society) and by successive refinements (*i.e.,* nested sub-societies), it goes down to the point where agent roles and interactions can be identified; every refinement step has a dedicated documentation. I made the notion of *roles* central to the approach, by using them as a turning point between the macro-level of societal requirements, structures and interactions, and the micro-level of individual agents and their implementation. Just as it was the case for ontologies, the experience revealed a strong need for tools that cover and assist the design and deployment at each step of the lifecycle of a multi-agent system.

The obtained architecture is flexible enough to produce a range of foreseeable configurations and to accept new roles for additional functionality with a minimum rework. The architecture was implemented and tested providing a real proof of concept and feasibility. I discussed the argument saying that this implementation could have been done in objet or component programming by arguing that technically speaking it could have been done in byte code (since at the end of the day it runs in this format), but the whole interest of agents is to provide a high level of abstraction reducing the gap between o the one hand the conceptualisation of a distributed problem and its solution, and on the other hand, the technical specification and the programming primitives used for the implementation. It is all the more true since the organisational design approach adopted here was perfectly natural in the context of the organisational issues of knowledge management; we built:
- *artificial societies of intelligence* relying on semantic-level message-based communication to collaborate over the global management and capitalisation of corporate knowledge.
- *artificial individual intelligence*, situated and able to locally adapt to available resources, to interfaced users, etc.

However this is not plead in favour of agents as being a silver bullet; when implementing a conceptualisation, some entities are considered as objects and other as agents, thus a real complete framework should be both agent-oriented and object-oriented, *i.e.,* these two entities should be considered as first citizens existing in a multi-agent and multi-object environment.

I particularly showed the interest of *merging ontology and distributed artificial intelligence to provide distributed mechanisms managing distributed knowledge*. The problem I addressed by merging them was the knowledge distribution *i.e.,* both the spatial property of knowledge of being scattered about and the act of diffusing knowledge. I focused on the annotation storage allocation and query decomposition and distributed solving. Since they are linked to one another, I chose two complementary strategies to implement them:
- the allocation of an annotation is based on a contract-net where bids are the result of a distance between the semantic contribution of the annotation and the description of the semantic contribution of an archive to the content of the memory.
- the solving of a query exploits the description of the semantic contribution of an archive to the memory to allocate sub-tasks to the relevant agents in charge of the archives.

Conversely, I explained that, even if this aspect had not been addressed in CoMMA, ontologies are living objects, and over time their uses reveal new needs for knowledge acquisition and formalisation following a never-ending prototype lifecycle. To implement this lifecycle and the collective consensus maintenance, a multi-agent architecture could be interesting. The *ontology-agent couple* is also the key of social-level specifications and integration at semantic-level message-passing; both are necessary to ensure the loosely-coupled feature enabling the valuable separation of concerns.

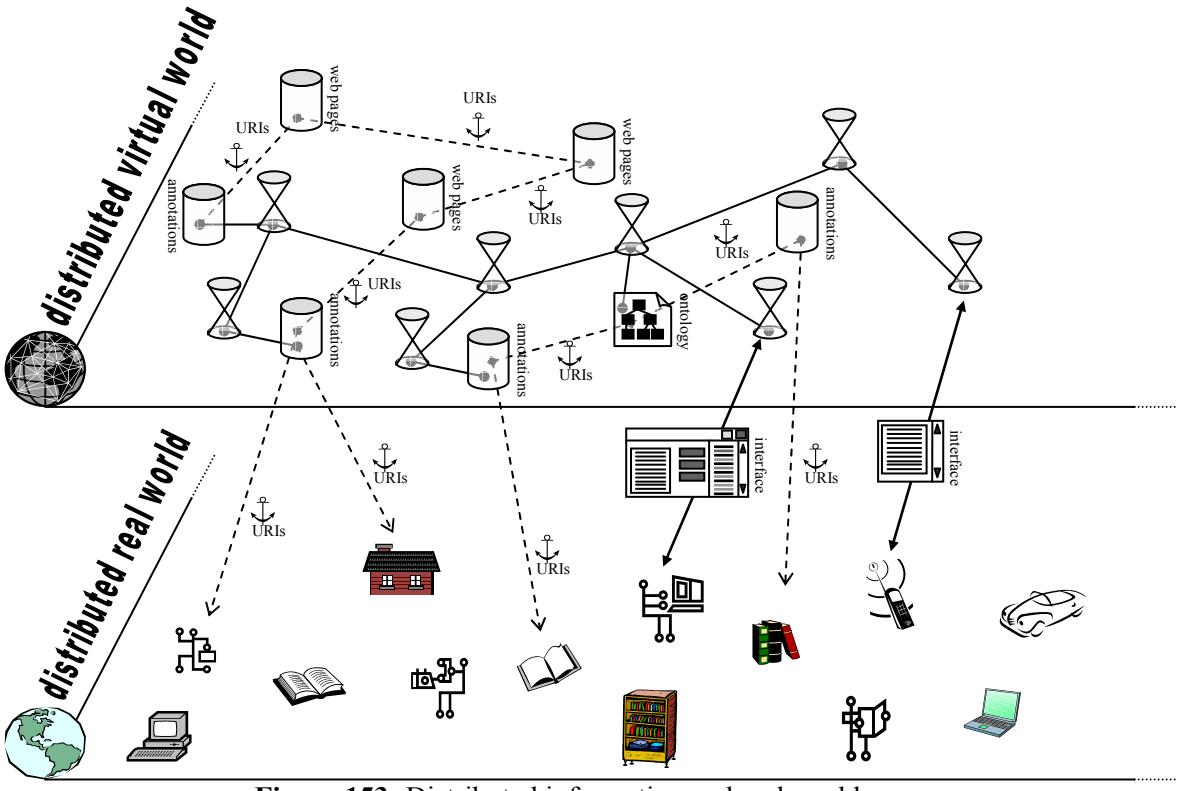


Figure 153 Distributed information and real worlds

Figure 153 summarises the overall vision I developed in this work; the one of a distributed virtual world maintaining a mapping with our real world to assist us in our day-to-day life.

The mapping exist:

- through *ontological aspects*: formal ontologies captures the semantics mobilised by human activities and provides modelling primitives. The ontologies are a mapping between the automated symbolic system we build and our conceptualisation of the world.
- through *assertional aspects*: URI-based annotations use the ontological primitives to represent the state of affairs of the real world (people, places, physical resources, etc.); by annotating the real world through URIs, they build a mapping between real entities and digital surrogates that represent them in the virtual world.
- through *deployment aspects*: the virtual world is distributed in our real world through networks of devices (computers, PDA, cellular phones, etc.) *i.e.,* a software entity is both situated in its virtual world and in our real world, its location is a mapping point between the two worlds. Exchanges occur at these mapping points though interfaces between these two worlds (a dedicated graphic user interface, a phone vocal interface, an e-mail etc.).

The virtual world provides digital information resources and is inhabited by artificial intelligent agents using the models (ontology and annotations) to manage this distributed landscape and be aware of the real world in their actions. Deliberative agents and formal knowledge form a natural symbiotic couple where formal knowledge allows artificial intelligent agents to act, and agents can take in charge the lifecycle of formal knowledge from creation to maintenance.

> *What is a Web year now, about three months? And when people can browse around, discover new things, and download them fast, when we all have agents, then Web years could slip by before human beings can notice.*
>
> — Tim Berners-Lee

CoMMA was a complete and complex experience. A *scenario-based approach was a good option both for agents and ontology-based corporate semantic web design*: for agents, scenarios initiated the organisational top-down functional analysis and provided a base to build use cases, interaction specifications, etc.; the for corporate semantic web scenarios captured the scope of the conceptualisation, focused knowledge acquisition, drove reuse prospecting and specified knowledge modelling. Scenarios implicate users in the design process and they were certainly an important reason for the good welcome of the last prototype, both by end-users and open-day visitors. The other important aspect in such a project is the ergonomic concerns for interfaces that will have to bridge the gap between the internal conceptual complexity and the stakeholders' day-to-day concerns. Presentation and representation means are a key need for user's acceptance.

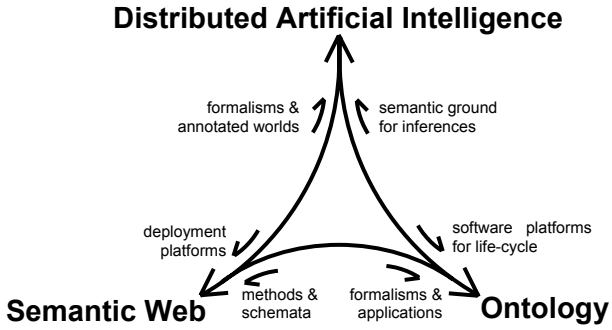


Figure 154 Complementarity triad of my research fields

Finally, I would like to underline that the fields chosen for my research are linked by a very strong complementarity:

- *Utility of ontologies for the agents*: the communication between agents is based on speech acts provided by an ontology, their inferences exploit semantic spaces defined by ontologies to support distributed mechanisms, etc.
- *Utility of ontologies for the semantic Web*: the realisation of the schemas defining the vocabulary of the annotations calls upon methods of ontology design, the new inferences of research in the bases of annotations exploit the properties of theontology, etc.
- *Utility of the agents for ontologies*: multi-agent architectures can be used for the realisation of collective software, for setting up and maintaining the ontological consensus, the agents can manage the gateways between various domains and their respective ontologies, etc.
- *Utility of the agents for the semantic Web*: the agents propose a suitable paradigm for search systems handling distributed annotation bases, the multi-agent architecture can also be used for the deployment of semantic Web services, etc.
- *Utility of the semantic Web for the agents*: the W3C proposes standardised languages for the Semantic Web which can be used as agent communication languages such as XML and RDF(S), the semantic web allows us to build annotated worlds where information agents can quickly become intelligent actors, etc.
- *Utility of the semantic Web for ontologies*: the W3C proposes standardised languages for the Semantic Web which can be used for the exchange and the extension of ontologies, the Semantic Web is also a perfect ground of application for ontologies, etc.

Thus, although it is not a conclusion in itself on the different problems I addressed in this PhD, my last remark is that the application of distributed artificial intelligence to the knowledge management problems is a real and challenging research field with a promising future that calls for a dedicated research and development community.

— Fabien Gandon

Part - V

Annexes

14 Résumé in french

Les sociétés humaines sont structurées et régies par des organisations. Les organisations peuvent être vues comme des organismes vivants, abstraits, composés d'individus et d'autres organisations. Leur raison d'être est un ensemble d'activités par lesquelles elles répondent aux besoins d'autres organisations ou individus. Ces activités centrales sont le résultat du travail collectif des membres de l'organisation. L'activité globale est basée sur les structures et les infrastructures de l'organisation, qui sont supervisées par les organes de gestion de l'organisation. La gestion vise à coordonner le travail des membres afin que, collectivement, ils réalisent les activités centrales de l'organisation.

Le travail individuel, qu'il fasse partie de la gestion ou des activités centrales de l'organisation, exige des connaissances. Toutes les connaissances mobilisées par une organisation pour son fonctionnement forme un ensemble abstrait appelé *les connaissances de l'organisation*; une carence en connaissances peut avoir pour conséquence un dysfonctionnement de l'organisation.

Alors que la vitesse d'évolution des marchés augmente et que leur dimension tend vers la globalisation, le temps de réaction permis diminue et la pression de la concurrence augmente; la perte d'informations peut se solder par une occasion manquée. Les organisations doivent réagir rapidement aux changements dans leur domaine et dans les besoins auxquels ils répondent, et même mieux: ils doivent les prévoir. Dans ce contexte, la connaissance devient une ressource capitale pour l'organisation, pour sa compétitivité et sa survie. Ainsi *la gestion de l'organisation inclut maintenant explicitement l'activité de la gestion de sa connaissance*, qui adresse les problèmes d'identification, d'acquisition, de mémorisation, d'accès, de diffusion, de réutilisation et de maintenance des connaissances internes et externes utiles à l'activité de l'organisation.

Une approche pour la gestion des connaissances dans une organisation, est le déploiement d'une solution de gestion de la mémoire organisationnelle: l'aspect *mémoire organisationnelle* est responsable d'assurer la persistance et l'indexation de la connaissance de l'organisation et la *solution de gestion* est responsable de capturer les connaissances appropriées et de les fournir aux personnes intéressées, les deux activités étant effectuées au moment opportun, avec un de détail et un format adéquats. Une mémoire organisationnelle se fonde sur des supports de connaissance *i.e.*, des

documents, des personnes, des connaissances formalisées (ex.: logiciels, bases de connaissance) et d'autres artefacts vecteurs de connaissances.

Pour être implantés, une telle mémoire et sa gestion exigent des méthodologies et les outils. Les ressources, telles que les documents, sont des supports de l'information donc, leur gestion peut donc utiliser les avancées de l'informatique et être assistée par des solutions logicielles. Le travail que je présente ici, a été effectué durant ma thèse in informatique au sein d'ACACIA, une équipe de recherche multidisciplinaire de l'INRIA, qui vise à offrir des modèles, des méthodes et des outils assistant le cycle de vie d'une mémoire organisationnelle. Mes recherches ont été appliquées dans le cadre du projet CoMMA (Corporate Memory Management through Agents) un projet IST Européen, de deux ans.

Dans cette thèse, je montre que (1) *les Web sémantiques peuvent fournir des espaces de connaissances distribués pour la gestion d'une mémoire organisationnelle*; (2) *que les ontologies sont des moyens appropriés et utilisables pour supporter ces espaces de connaissances distribuées*; (3) *que les systèmes multi-agent sont des architectures appropriées et utilisables pour gérer ces espaces de connaissances distribuées*.

14.1 Le projet CoMMA

La mémoire organisationnelle est une représentation persistante, explicite, désincarnée des connaissances et des informations dans une organisation, afin de faciliter leur accès, leur partage et leur réutilisation par les membres adéquats de l'organisation, dans le cadre de leurs tâches individuelles et collectives [Dieng *et al.*, 2001]. L'enjeu d'un système de gestion de mémoire d'entreprise est de réussir l'intégration cohérente de la connaissance dispersée dans l'organisation afin d'en promouvoir la croissance, la communication et la préservation [Steels, 1993]. La connaissance organisationnelle est naturellement distribuée au sens large du terme : elle n'est pas seulement distribuée entre des systèmes informatiques, mais inclut d'autres artefacts ainsi que les membres de l'organisation. Les solutions informatiques doivent prendre en compte cette super-distribution pour s'intégrer aux organisations. Les modèles de la distribution doivent aller au-delà de la dimension technique et rendre compte de l'organisation, de son infrastructure et de son environnement.

Une mémoire distribuée et hétérogène : la mémoire organisationnelle est, comme le Web, un paysage d'informations hétérogènes et distribuées. Tous les deux connaissent les problèmes de bruit et de précision lors de la recherche d'information : "Les utilisateurs du Web se noient dans l'information alors qu'ils ont soif de connaissance" (John Naisbitt). Parmi les initiatives visant à résoudre ces problèmes, le Web sémantique [Berners-Lee *et al.*, 2001] est une approche cherchant à rendre explicite la sémantique des documents par des annotations basées sur une ontologie. Les intranets reposant sur les technologies du Web peuvent bénéficier des progrès du Web sémantique en étudiant alors la mémoire organisationnelle d'une entreprise comme un web sémantique d'entreprise.

Une population d'utilisateurs distribuée et hétérogène : la population des utilisateurs est hétérogène car il existe plusieurs profils de membres concernés par la gestion et l'exploitation de la mémoire. Chacun a ses particularités, ses centres d'intérêts et son rôle dans l'organisation. Cette population est aussi distribuée puisque les utilisateurs sont dispersés dans l'infrastructure de l'organisation. Ils ignorent parfois l'existence, l'emplacement et la disponibilité d'artefacts et de gens avec lesquels ils partagent des centres d'intérêts. Pour s'adapter aux utilisateurs et aux contextes d'utilisation, l'apprentissage symbolique propose des techniques permettant d'apprendre les particularités individuelles ou d'assister l'émergence de communautés d'intérêt et la diffusion proactive d'information par une exploitation collective des profils individuels.

Une gestion distribuée et hétérogène: la mémoire et sa population d'utilisateurs étant distribuées et hétérogènes, il semble intéressant que l'interface entre ces deux mondes soit de même nature afin de pouvoir se calquer sur le paysage d'information et s'adapter aux utilisateurs. En programmation, les progrès se sont faits à travers le développement d'abstractions de plus en plus puissantes permettant de modéliser et de développer des systèmes de plus en plus complexes ; le paradigme des systèmes multi-agents (SMA) propose une nouvelle étape dans l'abstraction et peut être employé pour comprendre, modéliser, et développer des systèmes distribués complexes [Wooldrige *et al.*, 1999]. Ce paradigme apparaît adapté au déploiement d'une architecture logicielle au-dessus du paysage d'informations distribuées qu'est la mémoire d'entreprise: par leur collaboration, les agents réalisent une intégration des connaissances de l'entreprise et en permettent la capitalisation globale tout en s'adaptant localement à la spécificité des ressources et des utilisateurs.

Avec l'équipe ACACIA, j'ai participé au projet CoMMA - Corporate Memory Management through Agents [CoMMA, 2000] - qui visait à développer et tester un environnement de gestion de la mémoire d'entreprise pour deux scénarios :

(1) l'aide à l'insertion d'un nouvel employé en l'assistant dans ses recherches ou en lui suggérant proactivement les informations dont il a besoin pour s'intégrer dans l'organisation (2) le support à la veille technologique en assistant l'identification d'informations pertinentes pour l'activité de

l'entreprise et leur diffusion aux personnes concernées et compétentes. Pour cela, CoMMA intègre, au sein d'un SMA, des techniques et outils issus de l'ingénierie des connaissances, de la galaxie XML et de l'apprentissage symbolique - Figure 155.

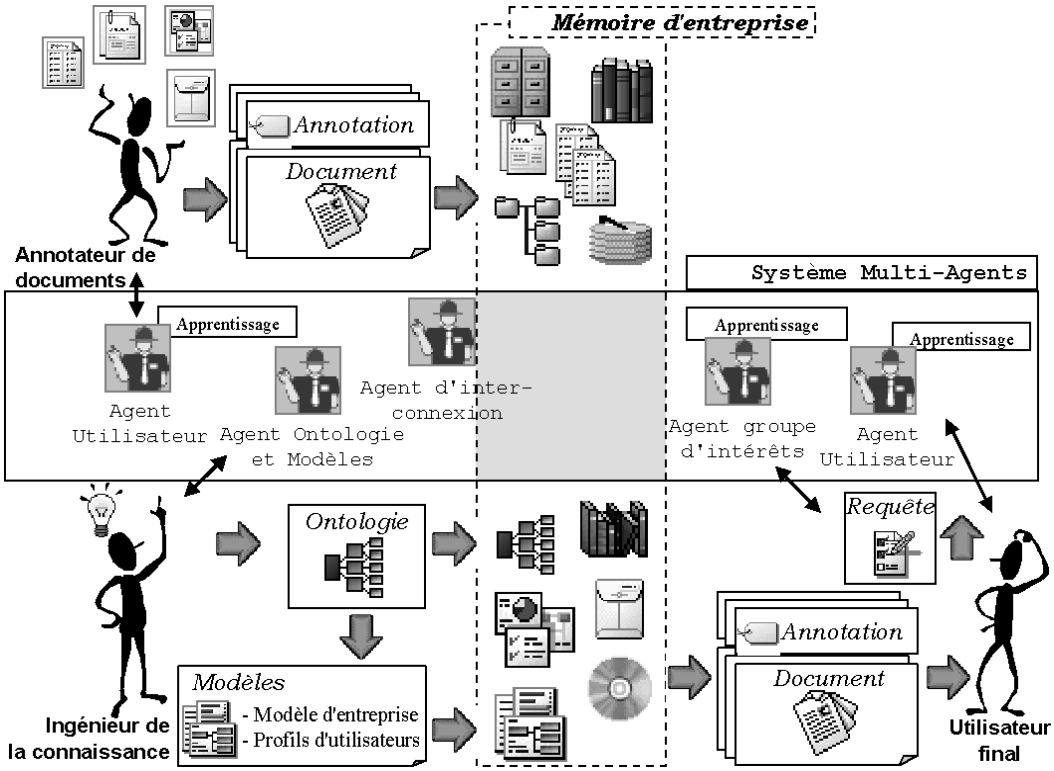


Figure 155 Schéma général de l'approche CoMMA.

14.2 Web sémantique d'entreprise

14.2.1 Du Web Sémantique à l'intraweb sémantique

La mémoire repose de plus en plus souvent sur un usage interne à l'organisation des technologies de l'Internet (les intranets) et du Web (les Webs internes ou intrawebs). Cela mène à des quantités de données et d'informations semi-structurées disponibles en ligne, mais enterrées et dormantes dans leur masse. Les mémoires organisationnelles ont maintenant à faire face aux mêmes problèmes que le Web en matière d'exhaustivité et de précision lors de la recherche d'information, et de façon général, en matière d'automatisation des tâches de gestion de cette mémoire.

Le Web sémantique est une approche prometteuse où la sémantique du contenu des documents est rendue explicite par des annotations utilisées pour guider une exploitation ultérieure, par exemple pour la recherche d'informations sur le Web. RDF (Resource Description Framework [Lassila and Swick, 1999] [Manola and Miller, 2002]) est à la base du Web sémantique et est doté d'une syntaxe XML. RDF nous permet d'annoter sémantiquement les ressources de la mémoire: il propose un modèle de données simple fournissant un langage de représentation des propriétés et des relations des ressources du Web. Nous étudions alors la mémoire d'une entreprise comme un Web sémantique d'entreprise ou intraweb sémantique: nous décrivons le contenu sémantique des documents de l'entreprise par des annotations sémantiques; ces annotations sont utilisées ensuite avec des inférences pour fouiller plus intelligemment la masse d'informations de la mémoire organisationnelle. De même qu'une caractéristique importante d'un nouveau logiciel est sa capacité à intégrer ou s'interfacer avec des systèmes existants, une caractéristique importante d'un système de gestion d'une mémoire organisationnelle est sa capacité à intégrer des archives existantes. Puisque les annotations RDF peuvent être internes ou externes aux documents, des documents existants de la mémoire peuvent être gardés intacts et dotés d'annotations externes. RDF ne fait aucune hypothèse sur le domaine d'application ; les annotations sont basées sur une ontologie formalisée en RDF Schema (RDFS) [Brickley and Guha, 2000] [Brickley and Guha, 2002]. Les étapes de notre approche sont : (1) Conception d'une ontologie qui sera formalisée dans un schéma RDFS. (2) La situation actuelle de l'organisation et de ses membres est décrite en utilisant les primitives de l'ontologie, fournissant au système un modèle de son environnement qu'il peut exploiter dans ses interactions avec les utilisateurs ; ces descriptions sont formalisées en RDF, en instanciant le schéma RDFS. (3) Les documents sont annotés pour structurer la mémoire ; les annotations instancient aussi le schéma RDFS formalisant l'ontologie et référencent les ressources documentaires et les objets de la situation actuelle si nécessaire, par exemple : "le rapport d'analyse de tendance 'nouvelles normes de la téléphonie mobile' a été écrit par la cellule de veille technologique du département D12 et concerne les sujets : UMTS, WAP".

14.2.2 Une mémoire basée sur un modèle

Les utilisateurs de la mémoire sont, par nature, hétérogènes dans leur profil et répartis dans l'organisation. Afin de donner au système un aperçu de son environnement et des utilisateurs avec lesquels il interagit, la mémoire est basée sur des modèles de la structure organisationnelle et sur des profils utilisateur. Ces modèles permettent la personnalisation et l'apprentissage des préférences des utilisateurs, et une diffusion intelligente et proactive de l'information, basée sur les centres d'intérêt des utilisateurs. Nous verrons que l'ontologie fournit les primitives pour la description des utilisateurs et de l'entreprise.

14.2.2.1 Description des profils utilisateur : "annoter les personnes"

Un profil d'utilisateur capture les aspects de l'utilisateur qui ont été identifiés comme pertinents (et formalisables) pour le comportement du système. Il contient des informations administratives et les préférences explicites qui vont de la personnalisation d'interface aux centres d'intérêt. Il positionne également l'utilisateur dans l'organisation (rôle, position, réseau de connaissance, etc.) permettant au système de cibler ses actions de diffusion. En outre, le système déduit des informations à partir d'une session d'utilisation: pour ce faire il mémorise l'historique des documents visités et, le cas échéant, l'évaluation effectuée par l'utilisateur sur la pertinence et la qualité des documents proposés. Grâce à l'exploitation de ces informations, le système apprend certaines préférences de l'utilisateur. Il les utilise ensuite pour la présentation de résultats ou pour la diffusion proactive d'informations. L'adaptation à l'utilisateur se fonde sur des techniques d'apprentissage symbolique [Kiss and Quinqueton, 2001] et le profil est matérialisé et échangé sous la forme d'une annotation RDF sur l'utilisateur concerné. Le vocabulaire conceptuel utilisé par cette annotation est donné par l'ontologie O'CoMMA.

14.2.2.2 Description de l'entreprise : "annoter l'organisation"

Un modèle d'entreprise est une représentation explicite et focalisée de l'organisation. Jusque très récemment, la modélisation d'entreprises était principalement destinée à la simulation et l'optimisation des systèmes de production. Elle fournissait des bancs d'essai pour les processus d'affaires et était utilisée pour leur rétroingénierie. Mais le changement des règles du jeu des marchés a amené les organisations à se rendre compte de la valeur de leur mémoire et du fait que les modèles d'organisation ont aussi un rôle à jouer dans cette application [Rolstadås, 2000].

Dans CoMMA, le modèle vise à supporter les activités de la mémoire d'entreprise survenant dans nos scénarios d'application. Ce modèle donne au système une vue de son contexte de fonctionnement et de l'environnement organisationnel. Ainsi, il peut exploiter les aspects décrits dans ce modèle pour l'interaction entre les agents et surtout entre les agents et les utilisateurs. Il est nécessaire, pour le système, d'avoir une compréhension de l'environnement qu'est l'organisation, de ses buts et de sa politique, de sorte que le système résultant fonctionne effectivement en collaboration avec les agents humains afin d'atteindre un objectif commun [Papazoglou and Heuvel, 1999].

Nous utilisons RDF pour implanter notre description de l'organisation, en annotant les entités organisationnelles (services, laboratoires, activités...) avec leurs relations (contrôle, utilise, est inclus, s'intéresse à, etc.). Là encore, les concepts et les relations utilisés sont définis par l'ontologie.

14.2.2.3 Architecture de la mémoire

Dans leur article à propos des agents dans des mondes virtuels annotés, [Doyle and Hayes-Roth, 1998] ont expliqué que les environnements annotés contenant des explications à propos des buts, des utilisations et des activités possibles dans les mondes où ils évoluent, permettent à des agents de devenir très rapidement des acteurs intelligents dans ces espaces. Les paysages d'information annotés permettent ainsi de rendre des agents d'information plus intelligents. Si la mémoire organisationnelle devient un monde annoté, les agents peuvent utiliser la sémantique des annotations et par des inférences, assister les utilisateurs dans leur exploitation de la mémoire d'entreprise.

Avec RDF(S), nous formalisons une ontologie, nous décrivons la teneur des documents et des descriptions de l'environnement organisationnel à travers des annotations sémantiques en RDF (cf. Figure 156) puis nous utilisons et nous inférons à partir de ces annotations pour intelligemment explorer la masse d'informations de la mémoire.

Cette approche amène la mémoire d'entreprise à être matérialisée comme un Web sémantique d'entreprise ou intraweb sémantique. Les vignettes de la Figure 157 décrivent et commentent une telle structure. Le cas de l'ontologie est présenté en détail dans la section suivante.

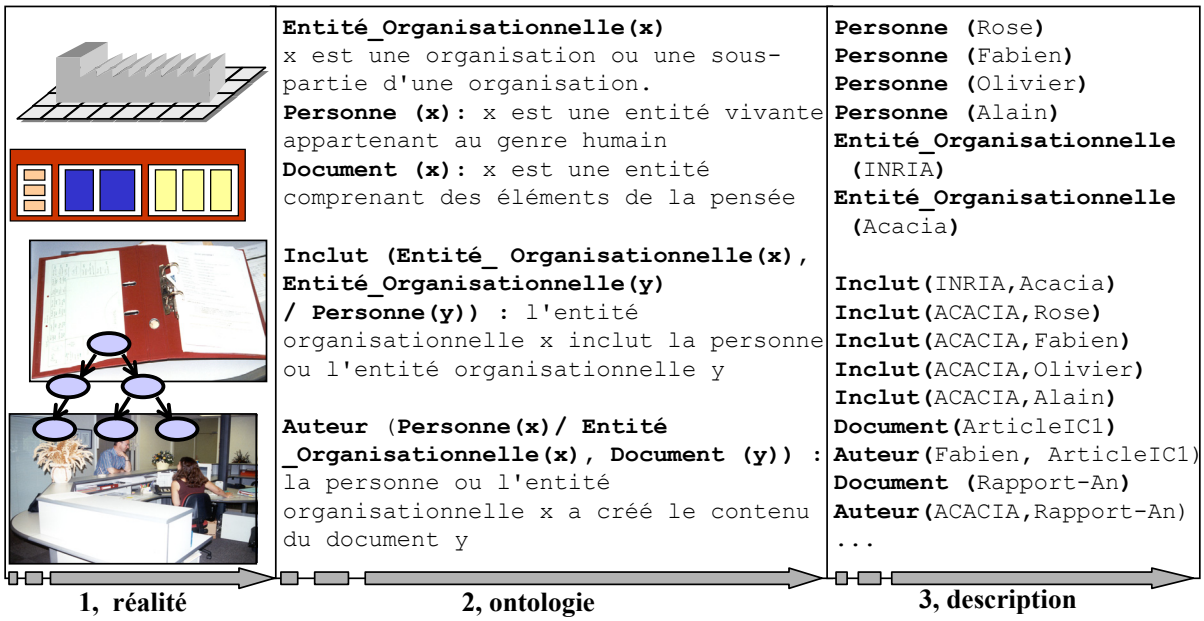
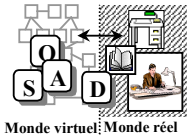
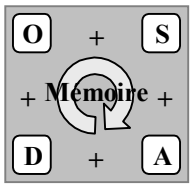


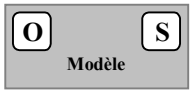
Figure 156 Etapes de modélisation et structuration



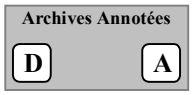
L'**O**ntologie, les descriptions de la **S**ituation et les **A**nnotations forment un monde virtuel capturant les aspects pertinents du monde réel avec lequel le système l'interagit .



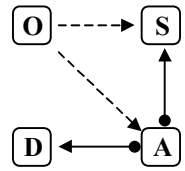
La mémoire se compose des **D**ocuments, de leurs **A**nnotations, des descriptions de la **S**ituation (utilisateurs et organisation) et de l'**O**ntologie. L'ensemble suit un cycle de vie prototypique.



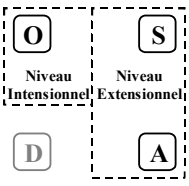
L'**O**ntologie et les descriptions de la **S**ituation forment le modèle, base de la structuration de la mémoire.



La structure des archives repose sur les **A**nnotations des ressources **D**ocumentaires.



Les **A**nnotations, et la description de la **S**ituation sont formalisées en utilisant le vocabulaire conceptuel de l'**O**ntologie. Les **A**nnotations référencent les **D**ocuments et les objets figurant dans les descriptions de la **S**ituation.



L'**O**ntologie capture les notions mobilisées pour nos scénarios et les articule au niveau intensionnel. Elle fournit les primitives pour les descriptions de la **S**ituation et les **A**nnotations qui s'inscrivent au niveau Extensionnel.

Figure 157 La structure de la mémoire

14.3 CORESE: moteur de recherche sémantique

Les moteurs de recherche par mots-clefs sont limités aux termes, partie visible mais ambiguë, dénotant l'utilisation des notions. L'introduction des ontologies permet aux logiciels de raisonner au niveau sémantique. Afin de manipuler l'ontologie et les annotations, l'équipe ACACIA a développé CORESE [Corby *et al.*, 2000] [Corby and Faron-Zucker, 2002] un prototype de moteur de recherche permettant de faire des inférences sur des annotations RDF. CORESE combine les avantages d'utiliser le modèle RDF(S) et sa syntaxe XML pour exprimer et échanger des méta-données, et les mécanismes de requête et d'inférence disponibles pour le formalisme des Graphes Conceptuels (GCs) [Sowa, 1984]. Il existe une véritable adéquation entre les deux modèles: les annotations RDF sont traduites en graphes-faits dans les GCs en n'utilisant que des relations binaires ; la hiérarchie des classes et celle des propriétés décrites dans un schéma RDFS sont traduites en une hiérarchie de types de concepts et une hiérarchie de types de relation dans le formalisme des GCs. Si nécessaire, un moteur d'inférence exploite des règles de pour compléter la base d'annotations avec des faits implicites déductibles.

Une requête CORESE est un énoncé RDF utilisant des variables pour décrire les énoncés recherchés, les valeurs à retourner et les liens de co-références contraignant ces valeurs ; les expressions régulières sont utilisées pour contraindre les valeurs littérales et des opérateurs supplémentaires sont utilisés pour exprimer la disjonction et la négation. La requête RDF est traduite en un graphe requête que l'on projette sur la base des graphes-faits. On isole des graphes correspondants pour en extraire les valeurs demandées; ces graphes réponse sont alors traduits de nouveau en RDF (Figure 158).

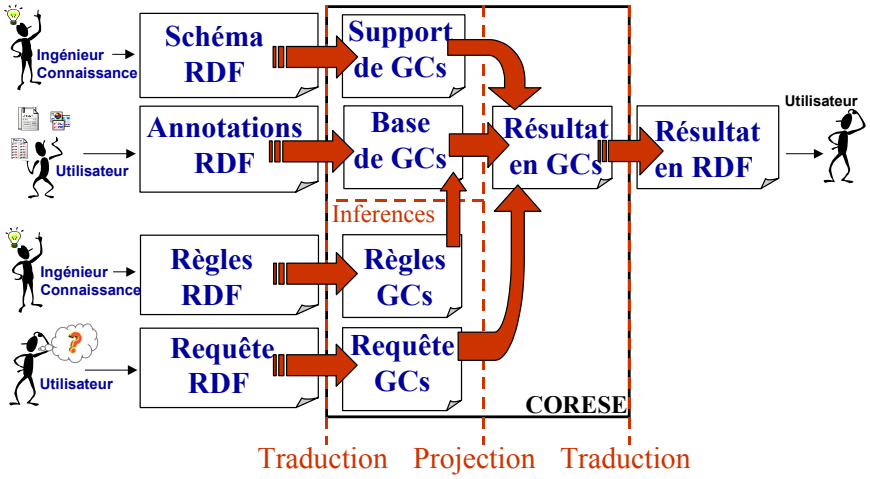


Figure 158 Principe de CORESE

Le mécanisme de projection tient compte des liens de spécialisation décrits dans les hiérarchies traduites à partir du schéma. Ainsi le moteur de recherche exploite les connaissances ontologiques pour améliorer le taux de rappel de sa recherche, par exemple: si un utilisateur demande les documents concernant l'intelligence artificielle et que le système connaît un document concernant les systèmes multi-agents, il sera capable d'utiliser l'ontologie pour inférer que les systèmes multi-agents sont une branche de l'intelligence artificielle et donc que ce document est une réponse pertinente.

Bien que CORESE puisse être utilisé en mode client-serveur, il offre également une API ; ainsi, comme nous le verrons dans la dernière section, des modules de CORESE sont intégrés dans les agents dédiés à l'ontologie, aux modèles et aux documents, afin de leur procurer les compétences nécessaires pour leurs rôles.

14.4 Conception De L'ontologie O'CoMMA

Il existe plusieurs sources de besoin d'une ontologie dans CoMMA : une venant de la nature même des SMA où les agents nécessitent un vocabulaire conceptuel consensuel pour exprimer les messages qu'ils échangent ; une autre venant de l'aspect système d'information utilisant des requêtes et des annotations basées sur un vocabulaire conceptuel consensuel ; enfin l'ontologie est un composant de la mémoire d'entreprise qui capture une connaissance potentiellement intéressante en elle-même pour l'utilisateur. Dans CoMMA, nous ne développons qu'une seule ontologie mais les différents agents et les différents utilisateurs n'exploitent pas les mêmes aspects de cette ontologie. A la différence du Web, une mémoire organisationnelle a un contexte délimité et mieux défini : l'organisation. Nous pouvons identifier les différents profils d'utilisateurs et comme la communauté de l'entreprise partage souvent un certain nombre de points de vue (politique d'entreprise, pratiques instituées, etc.), un consensus ontologique est possible dans une certaine mesure.

14.4.1 Position et Définitions

Un concept est un constituant de la pensée sémantiquement évaluable et communicable. L'ensemble des attributs d'un concept s'appelle sa compréhension ou son intension et l'ensemble des êtres qu'il englobe, son extension. Il existe une dualité entre l'intension et l'extension : si deux intensions vérifient $I1 \subset I2$, les extensions correspondantes vérifient $E1 \supset E2$. L'intension est déterminée par l'identification de propriétés communes à tous les individus auxquels le concept s'applique, cet ensemble de caractères permettant de définir ce concept.

Pour exprimer et communiquer l'intension, nous choisissons une représentation symbolique, presque toujours verbale, ex: les définitions des différentes notions associées à un terme dans un dictionnaire. Dans notre contexte d'un système d'information déployé dans une organisation humaine, une notion, pour être communicable, doit être dotée d'un libellé. Il est donc vital d'expliciter et de maintenir les liens entre les intensions capturées dans l'ontologie, les termes employés et les définitions associées en langue naturelle. C'est dans ces liens que se détectent l'ambiguïté et la synonymie. Notons que les exemples et les illustrations utilisés dans un dictionnaire montrent qu'il est parfois nécessaire, pour clarifier une définition en langue naturelle, d'exhiber un échantillon représentatif de son extension ou d'utiliser d'autres formes de représentations.

Les représentations des intensions sont organisées, structurées et contraintes pour exprimer une théorie logique rendant compte des relations qui existent entre les concepts. L'ontologie est cette théorie logique qui rend compte partiellement mais explicitement d'une conceptualisation, c'est-à-dire une structure sémantique intensionnelle qui capture les règles implicites contraignant la structure d'un morceau de réalité [Guarino et Giaretta, 1995]. L'ontologie est une représentation explicite partielle parce qu'elle se concentre sur les aspects de la conceptualisation nécessaires à l'application envisagée. L'ontologie est l'objet capturant les expressions des intensions et la théorie visant à rendre compte des aspects de la réalité choisis pour leur pertinence dans les scénarios considérés pour le système envisagé.

La représentation des intensions et de la structure ontologique peut faire appel à des langages plus ou moins formels selon l'opérationalisation envisagée pour l'ontologie. La construction formelle de l'intension donne une représentation précise et non ambiguë de la manière dont on peut concevoir son sens, ce qui permet sa manipulation logicielle et son utilisation comme une primitive de représentation de connaissances pour les descriptions et les annotations. L'expression de l'intension partant presque systématiquement d'une définition en langue naturelle, définir une ontologie est une tâche de modélisation menée à partir de l'expression linguistique des connaissances [Bachimont,

2000].Par raffinements successifs, nous augmentons l'ontologie en développant les pendants formels des aspects sémantiques pertinents pour le système dans les scénarios envisagés.

Dans l'ontologie, les concepts en intension sont habituellement organisés en taxonomie, c'est à dire une classification basée sur leurs similitudes. Lorsque des personnes s'accordent sur la théorie spécifiée par l'ontologie et sur son utilisation, nous disons qu'elles prennent un engagement ontologique. L'ingénierie ontologique s'occupe des aspects pratiques (méthodologies et outils) de l'application des résultats de la science ontologique à la construction d'ontologies, pour un cas précis et avec un but spécifique. Il s'agit donc d'un exemple d'ingénierie des connaissances.

14.4.2 Analyse par scénarios et Recueil

Concevoir l'ontologie, c'est d'abord identifier les notions du monde que l'on veut représenter. Ces notions étant accessibles en particulier au travers du langage, il s'agit de recueillir et d'analyser des corpus langagiers en étant guidés par des scénarios d'utilisation. Les corpus que nous avons recueillis et analysés sont de deux types : des corpus propres à l'entreprise et des corpus non spécifiques à l'entreprise - Figure 159 (1).

14.4.3 Les scénarios comme guides de conception

Les scénarios d'utilisation sont des descriptions textuelles de l'activité organisationnelle. Ils sont une façon naturelle et efficace de capturer les besoins des utilisateurs dans leur contexte [Caroll, 1997], ce qui est essentiel pour une intégration symbiotique du système à l'environnement de travail. Ils permettent : (a) de se focaliser sur les aspects de gestion des connaissances spécifiques au cas d'application considéré, capturant ainsi la portée du travail effectué ; (b) de capturer et présenter une image globale permettant de voir le système comme un composant d'une solution possible pour la gestion de connaissances dans une organisation ; (c) de rassembler un ensemble concret d'interactions avec la mémoire d'entreprise, compréhensible et accessible pour tous les intéressés du projet et fournissant un excellent point de départ pour construire des 'use case' ; (d) d'avoir un cadre pour évaluer et contrôler chaque nouvelle idée.

L'analyse par scénario nous a amenés à définir une grille (cf. Table 10 page 216) suggérant les principaux aspects à considérer lors de la description d'un scénario. Durant la phase de recueil, cette grille rappelle les informations à rechercher et les points qui pourraient être affinés. Les scénarios aident à définir le domaine de l'ontologie car elle doit fournir l'expressivité nécessaire aux interactions qu'ils décrivent. Les rapports de scénarios sont extrêmement riches et donc de très bons candidats à inclure dans le corpus de l'étude terminologique effectuée pour construire l'ontologie. Ils sont le résultat de notre première étape de recueil.

14.4.4 Recueil spécifique à l'entreprise

Plusieurs techniques existent pour le recueil de données, héritées des recherches effectuées en acquisition des connaissances. Ces techniques alimentent le processus de modélisation incluant la construction de l'ontologie. Nous avons essentiellement utilisé trois de ces techniques : les entretiens semi-structurés, l'observation et l'analyse de documents.

Un *entretien* peut être extrêmement structuré (interrogation) ou complètement libre (expression spontanée) mais le type le plus commun est l'entretien semi-structuré qui se trouve n'importe où sur le continuum entre ces deux extrêmes. Nos entretiens semi-structurés s'organisent ainsi : (a) première partie non structurée : une discussion lancée par des questions très ouvertes et générales concernant habituellement les tâches et le rôle des personnes interviewées. Si nécessaire, cette expression spontanée est encouragée par de courtes questions. (b) retour en arrière et clarification : une fois arrivé à un silence terminal, une partie plus structurée peut commencer nécessitant d'avoir pris des notes pendant la première partie et d'avoir déjà établi une certaine représentation personnelle afin de pouvoir identifier les points à clarifier ou détailler. C'est également dans cette partie que viennent les

questions plus spécifiques préparées avant l'entretien et motivées par des informations recherchées. (c) auto synthèse : une bonne pratique consiste à demander aux personnes interviewées de synthétiser, récapituler et analyser ce qu'elles ont dit pendant l'entretien et de les faire conclure. La généralisation et les regroupements qu'elles opèrent alors sont particulièrement intéressants pour structurer l'ontologie. Nous avons effectué un entretien avec une nouvelle employée, jeune secrétaire, de la société ATOS. L'entretien enregistré, retranscrit et exploité par mon collègue Alain Giboin a, par exemple, montré que la perception qu'une personne a de son rôle dans l'entreprise peut être sensiblement différente de la définition officielle donnée par sa fiche de poste (importance du profil personnel). Nous avons aussi trouvé des exemples de définitions de tâches liées aux documents (ex : mise en signature), ainsi que l'importance donnée au réseau de connaissances dans son activité quotidienne (utilité de la description de la structure organisationnelle).

La technique de *l'observation* s'intéresse aux personnes, la façon dont elles travaillent, sur une tâche réelle ou sur un scénario simulé, en temps réel, enregistré ou basé sur des traces de l'activité. Elle peut également se concentrer sur des indicateurs choisis (documents manipulés, organisation du bureau, réseau de connaissances, etc.). Selon l'acteur et le scénario, la situation intéressante peut être très différente (un nouveau venu recherchant une information, un mentor expliquant, une personne de la veille technologique commentant un nouveau brevet, etc.). Nous avons observé le lieu de travail de la jeune secrétaire par laquelle passe un grand nombre de documents internes. En s'appropriant son espace de travail, elle a réorganisé tous les documents et introduit certaines classifications. A titre d'exemple, elle a l'habitude d'utiliser deux critères pour étiqueter son bac 'documents en cours' : le type des documents (ex : formulaires de congés) et la prochaine tâche à faire sur ces documents (ex: à signer par le directeur). Les 'post-it' révèlent différents types et utilisations d'annotations sur des documents (ex: destinataire, date limite, mode d'utilisation d'un formulaire, autres documents liés). Enfin nous avons également noté qu'il existe des documents (ex: une carte de visite de la compagnie avec le numéro de téléphone et le numéro de fax) qu'elle ne souhaite pas trier ou ranger à part avec les autres (ex: annuaire téléphonique, carnet d'adresses) car elle veut pouvoir y accéder instantanément, au premier coup d'œil, en cas de besoin. On peut en déduire qu'un système doit pouvoir en offrir l'accès "au premier clic de souris".

L'analyse de documents typiques est une observation essentielle dans notre cas, étant donné que le projet se concentre sur la recherche d'informations dans une mémoire documentaire d'entreprise. Cette analyse suppose une collecte systématique des documents pertinents pour les scénarios considérés. Cela inclut les documents textuels (lexiques, rapports, etc.) qui peuvent faire l'objet d'une analyse de corpus par des outils de traitement de la langue naturelle, mais également les documents graphiques (formulaires, organigrammes, etc.) exigeant une étude manuelle. La collecte des formulaires vides puis remplis permet de voir leur utilisation, et leur suivi permet de découvrir leur cheminement dans l'organisation, et donc les flux de documents. Par exemple, "la fiche d'itinéraire du nouvel employé" d'un de nos partenaires industriels décrit pour un nouvel employé, ce qu'il doit faire, où aller, qui contacter, comment contacter et dans quel ordre effectuer ces tâches. Ce document indique les personnes impliquées dans le scénario d'insertion d'un nouvel employé, décrit le processus d'intégration et précise le vocabulaire requis pour décrire les différentes situations ou documents impliqués.

Ces recueils aident à comprendre quels types d'annotations peuvent exister et révèlent tout un vocabulaire lié à l'utilisation et l'organisation de la mémoire documentaire ; cependant, à l'exception de l'analyse de corpus textuels assistée par des outils linguistiques, ces recueils sont très consommateurs en temps, ce qui empêche de répéter ces expériences pour faire une véritable généralisation.

14.4.5 Recueil non spécifique à l'entreprise

Le recueil de données inclut également la réutilisation, la fusion et l'intégration d'ontologies existantes. J'ai étudié plusieurs ontologies : Ontologie d'Entreprise [Uschold *et al.*, 1998], Ontologie de TOVE [TOVE, 2000], Ontologie Supérieure de Cyc® [Cyc, 2000], Ontologie PME [Kassel *et al.*, 2000], Ontologie de WebKB [Martin et Eklund, 2000]). La réutilisation d'ontologies est à la fois séduisante (elle permet une économie de temps et d'efforts et favorise la normalisation) et difficile (il faut réajuster les engagements et les conceptualisations de l'ontologie réutilisée pour la nouvelle ontologie). Ces ontologies n'ont pas été importées directement ni traduites automatiquement. La meilleure façon de les réutiliser était d'analyser leur version informelle comme un document recueilli. Les différences entre les objectifs et les contextes de modélisation et d'utilisation de ces ontologies et ceux de notre ontologie, ainsi que les divergences dans la façon d'envisager une ontologie, m'ont amenés à penser qu'aucune importation automatique n'était possible et que la supervision humaine était obligatoire. Cependant, les outils de traitement de la langue naturelle tels que ceux étudiés par la communauté TIA [Aussenac-Gilles *et al.*, 2000] pourraient aider l'analyse et l'examen de la version en langue naturelle, et des traducteurs entre les différents formalismes de représentation des ontologies permettraient d'extraire plus rapidement les pans identifiés comme intéressants et directement réutilisables.

J'ai aussi enrichi ces différentes contributions en considérant d'autres sources informelles fournissant une expertise qui n'est explicitée dans aucun des recueils précédents. Des sources très générales m'ont aidés à structurer les parties supérieures de certaines branches de l'ontologie. Par exemple, des idées issues du livre "Using Language" d'Herbert H. Clark ont été utilisées pour structurer la branche des documents concernant les systèmes de représentation et pour injecter des réflexions issues de l'expertise sémiotique telles que la différentiation des systèmes de représentation iconique, indicielle et symbolique. D'autres sources très spécifiques m'ont permis de gagner du temps sur l'énumération de certaines feuilles de la hiérarchie taxinomique. Par exemple, la norme MIME était une excellente source pour la description des formats électroniques et le Dublin Core a permis d'amorcer l'identification des propriétés générales d'un document. Ces recueils rappellent que l'entreprise est une société dans une société, une culture dans une culture et que l'ontologie d'entreprise inclut de fait des primitives non spécifiques.

L'utilisation systématique des dictionnaires ou des lexiques disponibles est une bonne pratique. En particulier, les méta-dictionnaires sont extrêmement utiles car ils permettent un accès simultané à tous les dictionnaires qu'ils répertorient, et facilitent ainsi la comparaison des différentes définitions et la construction de celle qui correspond à l'intension voulue. J'ai énormément utilisé OneLook[1] qui a permis d'établir les premières définitions en anglais.

Toutes les sources réutilisées ont été élaguées avant d'être intégrées dans l'ontologie pour supprimer les concepts jugés inutiles. Les scénarios sont très utiles pour un tel élagage car ils capturent la portée de notre travail et une vision partagée par les différents membres du consortium. Ils peuvent être employés pour décider si une primitive est utile ou non en fournissant des exemples de situations où l'utilité de la primitive est manifeste. Une primitive qui n'est jamais mobilisée dans nos scénarios, telle que la relation de 'propriété' présente dans l'Ontologie d'Entreprise de AIAI, est élaguée.

14.4.6 Phase terminologique

Chaque terme dénotant un concept jugé utile pour le déroulement d'un scénario est candidat à l'analyse terminologique. Il entraîne avec lui tous les termes dénotant des notions proches et ses synonymes connus. Ainsi si nous considérons les termes *document*, *rapport* et *compte rendu d'analyse de tendance technologique* impliqués dans les scénarios de veille technologique, leurs candidatures sont très liées. Le point de départ peut être le terme *document* du fait d'une étude de la

[1] www.onelook.com

partie haute d'ontologies existantes, ou le terme *rapport* apparu lors de l'entretien avec la jeune assistante ou enfin le terme *compte rendu d'analyse de tendance technologique* à la suite de l'observation d'une instance particulière de ce document. Les termes candidats sont rassemblés dans un jeu de tableaux, passant ainsi à une structure de recueil semi-informelle. J'ai proposé des définitions pour chacun de ces termes candidats. Cette première version a été présentée aux membres du consortium et des ajustements et des extensions de bas niveau ont été faits par les partenaires industriels, ex:

> "**Référent de Domaine :** Observateur responsable de l'expertise d'un domaine et qui est chargé de gérer un groupe d'observateurs contributeurs pour ce domaine."

L'étude terminologique est au cœur de l'ingénierie ontologique et le principal travail sur les termes identifiés est la production des définitions consensuelles qui expriment l'intension de chaque concept. Il existe trois cas de figure :

- *Un terme correspondant à une et une seule notion* : le terme non ambigu devient le libellé de la notion.
- *Plusieurs termes correspondant à une notion* : les termes sont des synonymes, on garde la liste des synonymes et le terme le plus communément utilisé devient le libellé de la notion.
- *Un terme correspondant à plusieurs notions* : il y a ambiguïté, le terme est noté comme étant ambigu. Les différentes expressions d'intension sont définies et des termes non ambigus (ex: termes composés) sont choisis comme libellés.

Etiqueter des concepts avec des termes est à la fois commode et dangereux. C'est une source importante de rechute d'ambiguïté où nous retombons rapidement et souvent inconsciemment dans l'ambiguïté en utilisant les libellés selon la définition que nous leur associons naturellement, et non pas selon la définition qui leur a été réellement associée au cours de l'engagement ontologique. Comme le remarquent [Uschold et Gruninger, 1996], une bonne pratique pour des vérifications ponctuelles durant le travail de modélisation est de remplacer les libellés par des identificateurs sans signification. Cependant, cela rend l'ontologie illisible, la langue naturelle restant le premier moyen d'accès aux concepts et aux connaissances. Si du point de vue du système, les concepts tirent leur signification de l'expression formelle de leur intension et de leur position dans l'ontologie, il n'en est pas de même du point de vue de l'utilisateur pour qui le libellé linguistique est la représentation privilégiée. La représentation explicite du niveau terminologique et l'existence de fonctionnalités de gestion de ce niveau dans les outils assistant l'ontologiste sont une réelle nécessité. Tout au long de l'évolution de l'ontologie, j'ai essayé de capturer et de maintenir les liens niveau terminologique - niveau intensionel.

14.4.7 Structuration : du semi-informel au semi-formel

Les concepts obtenus sont organisés en taxonomie. Les principes derrière cette structuration remontent à Aristote qui définissait une espèce en donnant son "genus" (genos) et son "differentia" (diaphora). Le "genus" est un genre plus général sous lequel l'espèce considérée se place. Le "differentia" est ce qui caractérise l'espèce dans ce genre. Ainsi j'ai commencé à regrouper des concepts premièrement de façon intuitive, puis en raffinant itérativement la structure suivant les principes aristotéliciens étendus donnés par [Bachimont, 2000]. Ces principes poussent à éliminer le multi-héritage au profit de la multi-instanciation. Ceci pose un problème avec l'introduction de concepts de type 'rôles', dans l'ontologie qui ont tendance, à cause de RDFS, à rendre le multi-heritage nécessaire lorsqu'un concept rassemble plusieurs rôles. Une solution serait d'expliciter des points de vue dans la taxonomie [Ribière, 1999] et de limiter l'application de ces principes étendus à l'intérieur d'un point de vue. Une approche est proposée dans [Kassel *et al.*, 2000] introduisant la notion d'axes sémantiques pour regrouper sous un genus les différentes dimensions ou natures de critères suivant lesquelles le différentia s'énonce. Les principes étendus de Bachimont peuvent alors s'appliquer pour des concepts issus des mêmes axes sémantiques. Il nous faudrait étudier comment une telle approche pourrait s'implanter en RDFS.

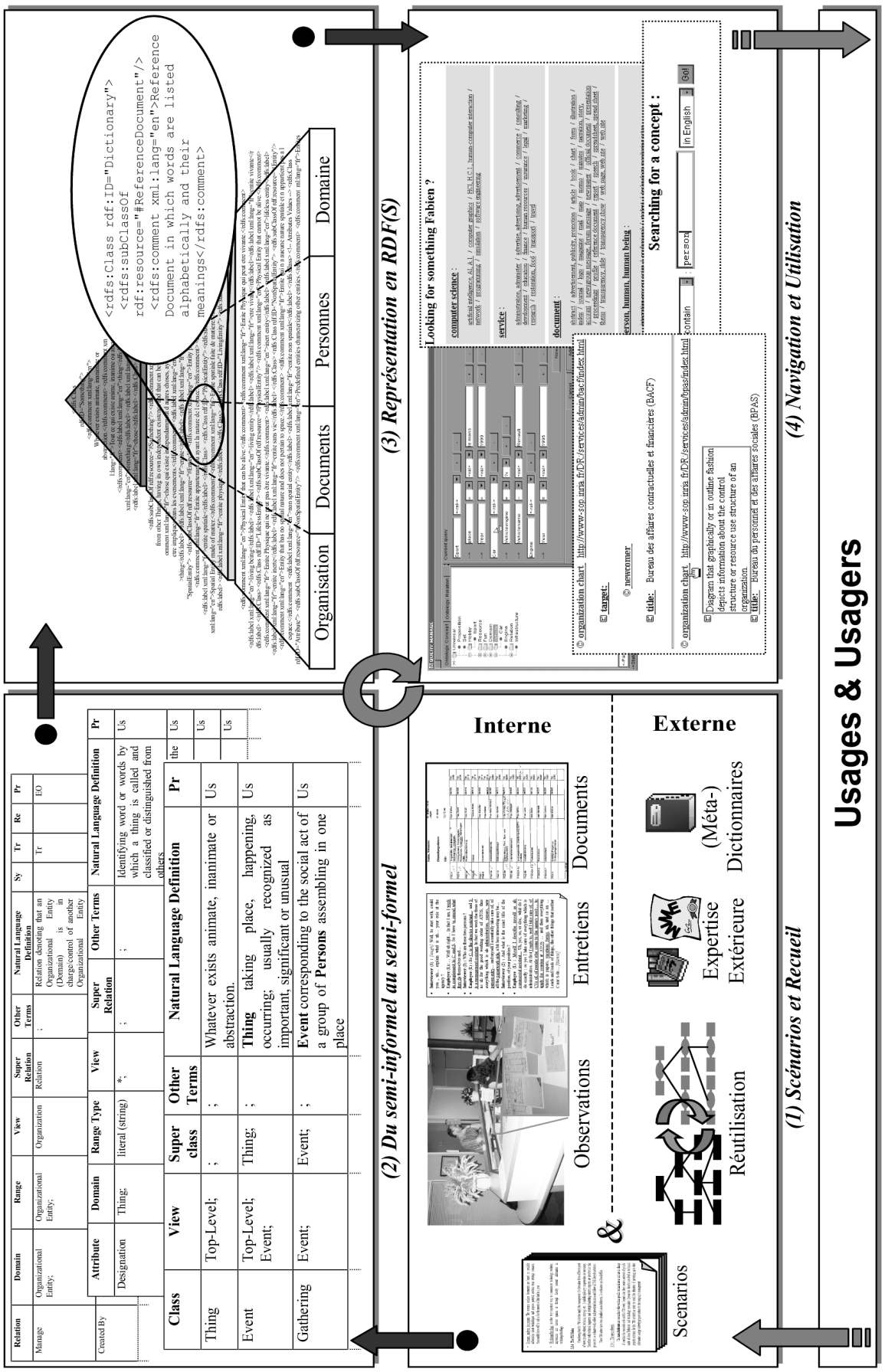


Figure 159 Les étapes de la construction d'O'CoMMA.

De même, le travail de Guarino et de Welty ([Guarino, 1992] ; [Guarino and Welty, 2000]) contribue à poser les bases fondamentales de l'ingénierie ontologique fournissant un cadre théorique, des définitions et des contraintes basées sur ces dernières permettant de vérifier la taxonomie : un ontologiste se fondant sur ces définitions peut ainsi contrôler certains aspects de validité de ses liens de subsomption. Cependant cela peut devenir un travail titanesque que d'appliquer ces théories à de grandes ontologies. Ces travaux semblent, actuellement, plus adaptés à la validation de la partie haute de l'ontologie qui, par extension, assure une cohérence minimale dans le reste de l'ontologie ; pour O'CoMMA ils ont servi à valider la couche haute, la plus abstraite, mais n'ont pas pu être mis en œuvre pour les autres couches.

La façon de concevoir une ontologie est encore sujette à beaucoup de discussions. Ma compréhension des différentes contributions faites jusqu'à maintenant est qu'il y a une tendance à distinguer trois options de construction :
- *Approche ascendante* : L'ontologie est construite par généralisation en partant des concepts des basses couches taxinomiques. Cette approche encourage la création d'ontologies spécifiques et adaptées.
- *Approche descendante* : L'ontologie est construite par spécialisation en partant de concepts des hautes couches taxinomiques. Cette approche encourage la réutilisation d'ontologies.
- *Approche centrifuge* : La priorité est donnée à l'identification de concepts centraux à l'application que l'on va ensuite généraliser et spécialiser pour compléter l'ontologie. Cette approche encourage l'émergence de domaines thématiques dans l'ontologie et favorise la modularité.

J'ai d'abord essayé une approche mixte (ascendante et descendante) alliant les atouts d'être spécifique et permettant de réutiliser d'autres ontologies. Après expérience, je ne suis pas convaincu qu'il existe réellement une approche purement ascendante, descendante ou centrifuge. Pour moi, il s'agit de trois perspectives complémentaires d'une méthodologie complète. Lors de la structuration taxinomique, la spécialisation d'un concept générique, ou la généralisation d'un concept spécifique sont présentes de façon concourante à chaque niveau de profondeur et à différents niveaux de granularité (au niveau des concepts, au niveau de groupes de concepts). La nature holistique de la connaissance implique la nature holistique des ontologies, et la nature holistique des ontologies mène à la nature holistique des méthodologies pour les établir. Je ne prétends pas que pour une application donnée, la construction ne se fera pas principalement selon une perspective (ex: des ontologies de substances chimiques font une utilisation intensive de l'approche ascendante). Cependant je ne pense pas qu'il faille opposer les différentes approches, mais plutôt considérer qu'elles correspondent à trois perspectives combinées en ingénierie ontologique : quand un ontologiste conçoit une ontologie, il effectue les tâches définies dans ces trois perspectives en parallèle.

Pour CoMMA, j'ai effectivement effectué certaines tâches en parallèle:
- En perspective descendante, j'ai étudié des ontologies de haut niveau et les couches hautes d'autres ontologies pour structurer la couche supérieure et réutiliser des distinctions existantes, ex: chose / entité / situation.
- En perspective centrifuge, j'ai étudié différentes branches d'autres ontologies ainsi que les principaux thèmes discernés pendant le recueil, pour circonscrire les principaux domaines nécessaires et regrouper leurs concepts.
- En perspective ascendante, j'ai exploité les rapports de l'analyse par scénarios et les traces du recueil pour identifier les concepts spécifiques et les regrouper par généralisation.

Les différents candidats dans chaque perspective (concept de haut niveau, central ou spécifique) sont à l'origine de sous-taxonomies partielles. L'objectif est alors de faire se rejoindre ces différentes sous-taxonomies partielles ; chaque approche influence les autres, un événement dans une perspective déclenche des vérifications et des tâches dans les autres.

A partir de la terminologie informelle, j'ai donc commencé à séparer les attributs et les relations des concepts et j'ai obtenu trois tableaux - Figure 159 (2). Ces tableaux ont évolué depuis une simple représentation semi-informelle (tableaux terminologiques terme/notion) vers une représentation semi-formelle (relations taxinomiques, signatures des relations) dont la structure finale était: le

libellé des concepts, relations ou attributs ; les concepts liés par les relations ou le concept et le type élémentaire liés par l'attribut ; le concept central le plus proche du concept considéré (personne, document...) ou les champs thématiques que la relation considérée relie ; les liens d'héritage du concept ou de la relation ; les termes synonymes du libellé ; une expression de l'intension en langue naturelle ; les propriétés de la relation : symétrique, anti-symétrique, transitive, réflexive ; la source de recueil ayant justifié l'ajout.

L'indication de la source de recueil permet la traçabilité des concepts, des relations ou des attributs et elle est intéressante en particulier pour abstraire une méthodologie du travail effectué. Elle permet de savoir quel type de contribution a influencé une partie donnée de l'ontologie et d'évaluer l'efficacité et l'impact de la réutilisation d'une source de recueil particulière. Cependant, cela ne suffit pas et la logique de conception ("design rationale") de l'ontologie devrait être capturée car elle explique ce qui a motivé sa forme actuelle ; ceci peut aider à la compréhension, l'adaptation et même l'adhésion à l'ontologie.

14.4.8 Formalisation de l'ontologie

Le travail de formalisation ne consiste pas à substituer une version formelle à une version informelle. Il s'agit d'augmenter une version informelle avec la correspondance formelle des aspects sémantiques intéressants et pertinents de l'ontologie informelle afin d'obtenir une ontologie documentée (description en langue naturelle éventuellement augmentée par des capacités de navigation introduites par la description formelle) et opérationnelle (description formelle des aspects sémantiques appropriés nécessaires aux opérations du système envisagé). L'ontologiste stoppe ce processus d'augmentation dès qu'il a atteint le niveau formel nécessaire et suffisant pour son système.

La version informelle de l'ontologie n'est donc pas une trace intermédiaire qui disparaît après la formalisation. L'ontologie formelle doit inclure les définitions en langue naturelle, les commentaires, les remarques, qui seront exploités par les personnes essayant de s'approprier l'ontologie. Les ontologies doivent être intelligibles aux ordinateurs comme aux humains [Mizoguchi and Ikeda, 1997]. Ceci joue un rôle important dans la réutilisabilité et la maintenance des ontologies. Les tableaux décrits précédemment ont évolué depuis une représentation semi-informelle vers une représentation semi-formelle par structuration itérative jusqu'à ce que la taxonomie soit suffisamment explicite pour être traduite. Un jeu de scripts utilisant les libellés non-ambigus nous a alors permis de basculer de la représentation semi-formelle des tableaux vers une représentation formelle en RDFS incluant tous les aspects de la représentation semi-informelle mais reposant sur la structure forte et explicite donnée par les relations taxinomiques - Figure 159 (3). Ce point de basculement ne change rien à l'ontologie en dehors de sa représentation interne.

La Figure 160 montre comment le formalisme RDF(S) est utilisé pour implanter les différents niveaux introduits précédemment :

- *Le niveau terminologique où s'organisent les termes recueillis*. Les relations entre le niveau intensionnel et le niveau terminologique dénotent les libellés possibles pour chaque intension (propriété `rdfs:label`). Une intension ayant des liens avec différents termes (ex. Figure 160 : I_4) est caractéristique de la synonymie de ces termes. Un terme ayant des liens avec différentes intensions (ex. Figure 160 : T_2) est caractéristique de l'ambiguïté de ce terme.
- *Le niveau intensionnel où s'inscrit la structure intensionnelle formalisant l'ontologie*. Les relations entre le niveau intensionnel et le niveau extensionnel représentent l'instanciation de chaque classe de concept. Les faisceaux de liens relient une intension à son extension (ex. Figure 160: I_8 et E_8).
- *Le niveau extensionnel où s'organise la mémoire factuelle* (les annotations des documents, la description de la situation organisationnelle et les profils utilisateurs). Une extension reliée à plusieurs intensions (ex. Figure 160: I_6 et I_7) est caractéristique de la multi-instanciation.

Figure 160 Formalisation en RDF(S)

Grâce aux feuilles de style XSLT, nous pouvons régénérer en permanence les vues informelles ou semi-formelles précédentes - Figure 159 (4) - notamment:
- Une feuille de style recrée le tableau terminologique initial, présentant un lexique de la mémoire.
- Un jeu de feuilles de style permet de rechercher des termes et de naviguer au niveau terminologique ou au niveau intensionnel en proposant en permanence de passer de l'un à l'autre. On peut ainsi naviguer dans la taxonomie des intensions de concepts ou de relations, désignées par les ensembles de libellés synonymes. On peut consulter les signatures des relations portant sur un concept et demander à voir l'extension d'une intension. Un échantillon d'extensions joue le rôle d'exemplification automatique et permet d'augmenter la compréhension du concept.
- Deux feuilles proposent respectivement un arbre indenté des intensions des concepts et des relations. Leur définition associée apparaît dans une fenêtre "popup" suivant la souris.

L'aspect terminologique permet une meilleure navigation ainsi qu'un accès multilingue à la structure des intensions et à la mémoire des extensions. Les feuilles de styles sont aussi utilisées pour documenter le résultat des requêtes en utilisant les définitions et les termes associés aux intensions. L'affichage des définitions dans de petites fenêtres "popup" est une première tentative pour désambigüer par anticipation la navigation ou les requêtes. Avant même que l'utilisateur ne clique, le système affiche sa définition en langue naturelle du concept sur lequel se trouve la souris ; cette fenêtre "popup" invite l'utilisateur à comparer sa définition personnelle avec la définition employée par le système pour éviter tout malentendu.

14.4.9 Extensions nécessaires à la formalisation

Le moteur CORESE a été étendu pour apporter une solution aux limitations d'expressivité qui ont été rencontrées lors de la formalisation de l'ontologie en RDFS; en particulier pour formaliser des connaissances de base ou connaissances implicites.

Par exemple, quand nous déclarons que quelqu'un dirige un groupe il est implicite que cette personne est un directeur. Ainsi le concept de 'directeur' devrait être un concept défini, c'est-à-dire un concept ayant une définition formelle explicite lui permettant d'être dérivé d'autres concepts existants. Cependant la notion de concept défini n'existe pas en RDFS.

Un deuxième exemple est la notion de 'collègue', collectée pour le scénario de veille technologique. Cette notion a d'abord été intégrée dans l'ontologie en tant que concept. Cependant, personne n'est un 'collègue' en soi. J'ai donc introduit une relation 'x est un collègue de y'. Cette relation nécessitait d'être typée 'relation symétrique', mais RDFS ne fournit pas cette primitive. Enfin, en considérant les scénarios d'application, il s'est avéré que cette relation ne serait pas utilisée pour l'annotation mais qu'elle serait inférée de ce qui est décrit de la structure organisationnelle (ex: M. Corby et Mme Dieng-Kuntz appartiennent au projet ACACIA donc il existe une relation 'collègue' entre M. Corby et Mme Dieng-Kuntz). J'avais donc besoin de pouvoir définir formellement la condition suffisante pour que deux personnes soient collègues et de pouvoir changer cette définition si nécessaire (par exemple pour une autre organisation). Ces limitations de RDFS deviennent rapidement un problème puisque la capacité à factoriser la connaissance dans une ontologie exige la possibilité de décrire des définitions formelles.

Dans la version actuelle de l'ontologie, les définitions formelles sont représentées dans des règles écrites dans un langage de règles RDF/XML spécialement créé pour RDF(S) et CORESE. Les caractéristiques de symétrie, de transitivité et de reflexivité des propriétés ont exigé des extensions de RDFS spécifiques à CORESE. Au lieu d'utiliser des règles nous pourrions également étendre le modèle de RDFS pour ajouter l'expressivité manquante comme le propose DRDF(S) [Delteil *et al.*, 2001], OIL [Fensel *et al.*, 2000], ou DAML+OIL[1]. L'exemple de la règle pour 'collège' est donné en Figure 56 page 240.

[1] http://www.daml.org/2000/12/daml+oil-index

14.5 Contenu de l'ontologie O'CoMMA

De l'approche décrite précédemment, a résulté l'ontologie O'CoMMA. Elle contient 470 types de concepts organisés en une hiérarchie d'une profondeur maximale de 13 liens de subsomption, 79 types de relations formant une hiérarchie d'une profondeur maximale de 2 liens de subsomption. O'CoM-MA utilise 715 termes anglais et 699 termes français pour étiqueter ces primitives ainsi que 547 définitions en Français et 550 définitions en Anglais. La montre Figure 161 les trois couches structurant l'ontologie:

- une couche supérieure très générale qui ressemble aux autres ontologies de haut niveau ;
- une couche médiane assez importante, divisée en une partie générique concernant le domaine des mémoires organisationnelles (documents, organisations, personnes...) et une autre consacrée au domaine d'application (ex. pour les télécommunications: WAP, mobile, etc.) ;
- une couche d'extensions spécifiques au scénario et à l'entreprise, avec des concepts complexes (ex. rapport d'analyse de tendance, carte d'itinéraire du nouvel employé).

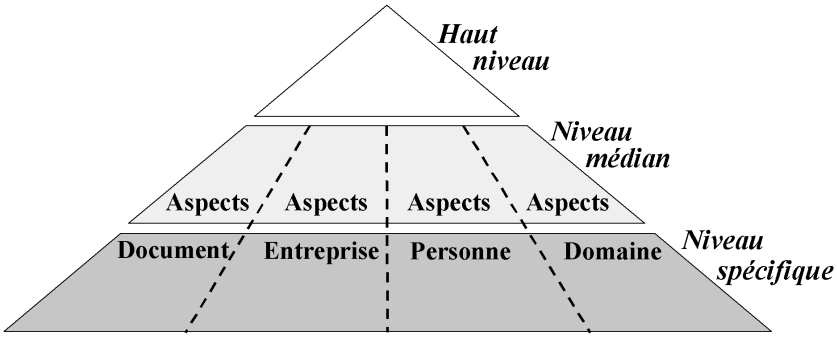


Figure 161 Structure de O'CoMMA

La partie supérieure, qui est très abstraite, et la première partie de la couche médiane, qui décrit des concepts communs aux applications de mémoire d'entreprise, sont réutilisables dans d'autres scénarios d'applications concernant la mémoire d'entreprise. La deuxième partie de la couche médiane, qui traite du domaine d'application, n'est réutilisable que pour des scénarios dans le même domaine d'application. La dernière couche contenant des concepts spécifiques n'est plus réutilisable dès lors que l'organisation, le scénario ou le domaine d'application change. Cependant, cette dernière couche est de loin la plus intéressante pour l'utilisateur car elle recèle les concepts qu'il manipule dans son travail quotidien.

Les partenaires industriels confrontés à l'ontologie trouvent sa complexité bien trop grande, ce qui la rend complètement hermétique. En particulier, le niveau supérieur de l'ontologie introduit des distinctions philosophiques tout à fait intéressantes d'un point de vue de la modélisation (le haut de l'ontologie fournit des briques saines permettant de démarrer la modélisation et d'assurer certaines vérifications de la cohérence) mais très hermétiques et souvent inutiles pour l'utilisateur typique du système. En outre, plus on monte dans la taxonomie, plus le consensus est difficile. Deux collègues de travail seront plus facilement d'accord sur la modélisation de concepts qu'ils manipulent et échangent tous les jours (ex: une news est un type de document présentant de nouvelles informations) que sur le haut de l'ontologie qui demande un grand niveau de généralisation et d'abstraction et porte sur des concepts que nous ne sommes pas habitués à discuter et manipuler tous les jours (ex: les choses se divisent entre les entités et les situations, les entités étant des choses capables de jouer un rôle dans une situation). Le haut niveau relève souvent d'une culture voire de croyances personnelles et ses concepts sont utiles pour des manipulations internes au système (généralisation de requêtes, structuration des couches hautes) mais pas pour l'interaction directe avec l'utilisateur. D'un point de vue ergonomique, nous avons étudié des interfaces filtrant l'ontologie en

fonction du profil de l'utilisateur ou proposant plusieurs modes de requête en fonction de l'expérience et du besoin de l'utilisateur.

Enfin, avec cette première version de l'ontologie, se posent maintenant les problèmes de son cycle de vie et de la maintenance, non seulement de l'ontologie mais aussi de ce qui a été construit au-dessus (annotations, modèles, inférences...). Il nous reste à étudier comment gérer ce cycle de vie global.

14.6 Les agents de la mémoire

Un web sémantique d'entreprise est un paysage d'informations hétérogènes et distribuées, annoté sémantiquement en utilisant les primitives fournies par l'ontologie. Pour gérer cet intraweb sémantique, il est intéressant d'envisager une architecture logicielle qui soit elle-même hétérogène et distribuée. L'adéquation des systèmes multi-agents a été reconnue dans un large spectre de projets proposant des systèmes multi-agents pour adresser différents aspects de la gestion des connaissances à l'intérieur des organismes. Ainsi, CASMIR [Berney and Ferneley, 1999] et RICOCHET [Bothorel and Thomas, 1999] se concentrent sur la collecte d'informations, l'adaptation des interactions aux préférences de l'utilisateur et l'apprentissage des centres d'intérêts afin d'établir des communautés et un filtrage collectif de l'information dans l'organisation. KnowWeb [Dzbor *et al.*, 2000] utilise les agents mobiles pour permettre l'accès à un réseau changeant dynamiquement, et exploite un modèle du domaine pour extraire les concepts représentatifs des documents afin de les utiliser pour répondre aux recherches de l'utilisateur. RICA [Aguirre *et al.*, 2000] maintient une taxonomie thématique des documents et l'emploie pour émettre des suggestions aux agents d'interface en se basant sur les profils des utilisateurs. FRODO [Van Elst and Abecker, 2001] est consacré aux mémoires distribuées avec une emphase particulière sur la gestion des ontologies du domaine.

CoMMA ne gère pas directement des documents, mais des annotations à propos de documents référencés par leur URI. CoMMA s'est focalisé sur trois fonctionnalités : (a) améliorer la précision et le rappel lors de la recherche de documents, en utilisant des annotations sémantiques ; (b) suggérer proactivement la consultation d'une ressource documentaire en utilisant des modèles de l'organisation et de l'utilisateur ; (c) l'archiver intelligemment les annotations soumises. L'architecture de CoMMA telle qu'elle était à la fin du projet sera détaillée dans la section suivante.

14.6.1 Une architecture multi-agents

Les tâches à exécuter sur la mémoire organisationnelle, la mémoire elle-même et la population des utilisateurs sont distribuées et hétérogènes. Par conséquent, il est intéressant que l'architecture logicielle du système soit elle-même hétérogène et distribuée.

Les progrès de la programmation ont été réalisés à travers des abstractions de plus en plus élevées nous permettant de modéliser des systèmes de plus en plus complexes. Les Systèmes Multi-Agents (SMA) sont candidats à être une nouvelle étape dans les niveaux d'abstraction, pour comprendre, modéliser et développer des systèmes répartis [Wooldridge *et al.*, 1999]. De plus, les SMA sont reconnus comme une architecture logicielle possible pour supporter le déploiement du Web sémantique [Berners-Lee *et al.*, 2001].

Dans CoMMA, le paradigme SMA est apparu très adapté au déploiement d'une architecture logicielle au-dessus du paysage d'informations distribuées qu'est la mémoire d'entreprise: d'une part, les différents agents s'adaptent localement aux utilisateurs et aux ressources auxquels ils sont dédiés ; d'autre part, grâce à leur coopération, les agents logiciels répartis sur l'intranet nous proposent une vue intégrée et globale de la mémoire organisationnelle matérialisée comme un intraweb sémantique.

14.6.1.1 Système d'information multi-agents

Les agents d'information sont une sous-catégorie des agents intelligents. Un SMA est un réseau d'agents faiblement couplés, qui fonctionnent ensemble comme une société visant à résoudre des problèmes qui seraient généralement au-delà des capacités de tout agent pris individuellement. Un SMA est hétérogène quand il inclut des agents d'au moins deux types. Un Système d'Information Multi-Agents (SIMA) est un SMA visant à fournir une partie ou une gamme complète de

fonctionnalités pour gérer et exploiter des ressources d'information. Lors d l'application des SIMA aux mémoires organisationnelles, la coopération des agents vise à améliorer la capitalisation de la connaissance dans l'organisation. L'architecture logicielle de CoMMA est ainsi celle d'un SIMA hétérogène dédié à la gestion d'un intraweb sémantique.

Une architecture SMA est une structure qui dépeint les différentes familles d'agents et leurs rapports. Une configuration est l'instanciation d'une architecture avec un agencement choisi et un nombre approprié d'agents de chaque type. Pour une architecture donnée, on peut générer plusieurs configurations, et une configuration donnée est étroitement liée à la topographie et au contexte de l'endroit où elle est déployée (structure de l'organisation, caractéristiques de l'Intranet, localisation des intéressés, etc.). Ainsi, l'architecture doit être conçue de sorte que l'ensemble des configurations possibles couvre les différents contextes organisationnels envisageables. La configuration est étudiée et documentée lors du déploiement alors que la description architecturale est étudiée et fixée lors de la conception.

14.6.1.2 Du niveau macroscopique du SMA au niveau microscopique des agents

L'analyse architecturale part du niveau d'abstraction le plus élevé (*i.e.* la société) et par raffinements successifs (*i.e.* sociétés imbriquées), elle descend jusqu'au point où les rôles et les interactions entre agents peuvent être identifiés.

Nous avons adopté une approche organisationnelle : l'architecture SMA est abordée, comme une société humaine, en termes de rôles et de rapports. Dans cette société artificielle, l'objectif commun est la gestion et la circulation des connaissances distribuées, rendues possibles par une ontologie commune et partagée.

Les spécifications fonctionnelles du système ne se calquent pas simplement sur des fonctionnalités d'agents mais ont une influence sur les interactions sociales des différents agents et sur l'ensemble des capacités, des rôles et des comportements qui leur sont attachés. En considérant les fonctionnalités du système, nous avons identifié trois sociétés d'agents dédiées aux ressources (l'ontologie et les modèles ; les annotations ; les pages jaunes pour gérer les interconnexions entre agents) et une dédiée aux utilisateurs (Figure 162).

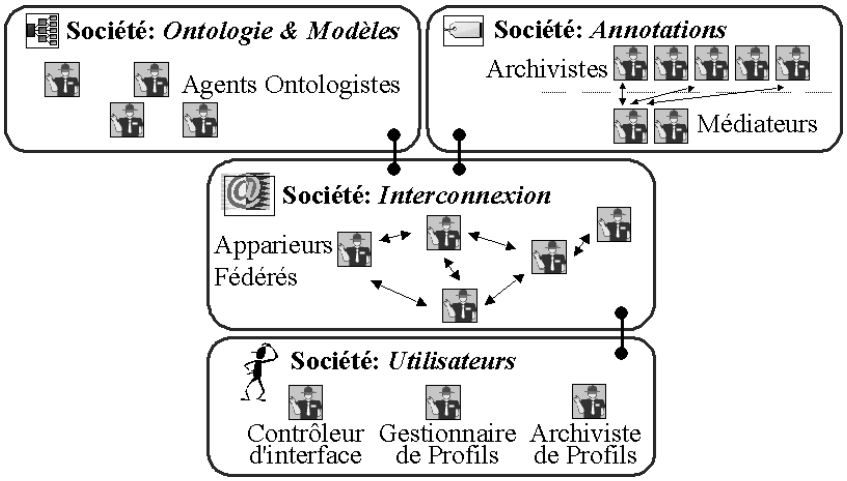


Figure 162 Sociétés pour la gestion de la mémoire

En analysant les sociétés dédiées aux ressources, nous avons identifié un ensemble récurrent d'organisations possibles: hiérarchique, égalitaire ou réplication. Selon le type de tâches devant être exécutées, et selon la taille et la complexité des ressources manipulées, nous avons préféré telle ou telle organisation de la société d'agents.

La société dédiée à l'ontologie et aux modèles est actuellement organisée comme une société de réplication : elle rassemble des agents ontologistes ayant chacun une copie complète de l'ontologie.

La société dédiée aux annotations est une organisation hiérarchique : des agents médiateurs d'annotations sont responsables d'allouer les nouvelles annotations et les tâches de résolution distribuée des requêtes aux agents archivistes d'annotations.

Les agents dédiés aux pages jaunes sont dans une organisation égalitaire : ils sont fournis par la plate-forme JADE [Bellifemine *et al.*, 2001] utilisée pour l'implantation de CoMMA. Les agents de cette société sont appelés 'apparieurs fédérables' : ils coopèrent pour apparier les agents demandeurs d'un service de la mémoire et les agents susceptibles de fournir un tel service. La description de ces services repose elle aussi sur une ontologie consensuelle aux développeurs.

Les agents de la société dédiée à l'utilisateur sont concernés par l'interface, la surveillance, l'aide et l'adaptation à l'utilisateur. Ils sont typiquement demandeurs de services et de ressources. Puisqu'ils ne sont pas liés à un type de ressource comme dans les sociétés précédentes, ils ne peuvent pas être étudiés en utilisant notre typologie. Nous pouvons cependant distinguer au moins deux rôles récurrents dans ce type de société:
- la gestion d'interface utilisateur: pour dialoguer avec les utilisateurs, leur permettre d'exprimer leurs re-quêtes puis de les raffiner, et présenter les résultats dans un format adéquat ;
- la gestion du profil utilisateur: pour archiver et rendre les profils disponibles afin qu'ils soient utilisables pour l'interface, les techniques d'apprentissage symbolique et les processus de diffusion proactive de l'information.

14.6.1.3 Rôles, interactions et protocoles

L'analyse de l'architecture nous a permis d'identifier des rôles d'agents et nous pouvons alors étudier leurs caractéristiques et leurs interactions pour pouvoir implanter les comportements correspondants dans un ensemble de types d'agent.

Les rôles représentent la position d'un agent dans une société, les responsabilités et les activités assignées à cette position et considérées par les autres agents comme étant remplies. Ces activités, couplées aux interactions, participent à la gestion de la mémoire distribuée. Chaque rôle identifié est décrit et avec ses caractéristiques après analyse et des quatre sociétés précédentes, dix rôles d'agent ont été identifiés et documentés:
- le contrôleur d'interface de (IC) contrôle et surveille l'interface utilisateur ; plusieurs interfaces ont été étudiées pour permettre à un utilisateur de manipuler l'ontologie lors des différentes tâches des scénarios d'application. L'interface finale de CoMMA (Figure 163) est un croisement entre plusieurs interfaces. L'utilisateur construit sa requête ou son annotation par parcours en Z de l'interface au cours duquel: (1) il donne un mot-clef (2) il choisit une notion parmi celles qui peuvent être dénotées par ce terme (3) il positionne cette notion dans la structure de son annotation/ sa requête (4) choisi des attributs littéraux et leur valeurs (ex. nom, titre, etc.).
- le gestionnaire de profils utilisateur (UPM) analyse les demandes et le retour des utilisateurs pour apprendre leurs préférences dans les résultats.
- l'archiviste des profils utilisateur (UPA) sauvegarde, fournit et questionne les profils des utilisateurs afin de répondre aux besoins des autres agents. Il compare également les nouvelles annotations et les profils d'utilisateur pour détecter les nouveaux documents intéressants pour un utilisateur et suggérer leu consultation.
- la gestion des agents (AMS) met à jour les pages blanches où les agents s'enregistrent et recherchent l'adresse d'autres agents à partir d'un nom ; il est fourni par JADE.
- l'apparieur (DF) met à jour les pages jaunes où les agents s'enregistrent et recherchent l'adresse d'autres agents à partir d'une description de services ; il est fourni par JADE.
- l'archiviste de l'ontologie de (OA) sauvegarde et fournit l'ontologie O'CoMMA au format RDFS ;
- l'archiviste du modèle d'entreprise (EMA) sauvegarde et fournit le modèle de l'organisation au format RDF.
- l'archiviste d'annotations (AA) sauvegarde et cherche les annotations RDF d'une archive locale à laquelle il est associé.

- le médiateur d'annotations (AM) décompose et distribue les tâches impliquées dans la résolution d'une requête et l'allocation d'une annotation, et notifie les agents qui souhaitent de la soumission d'une nouvelle annotation.

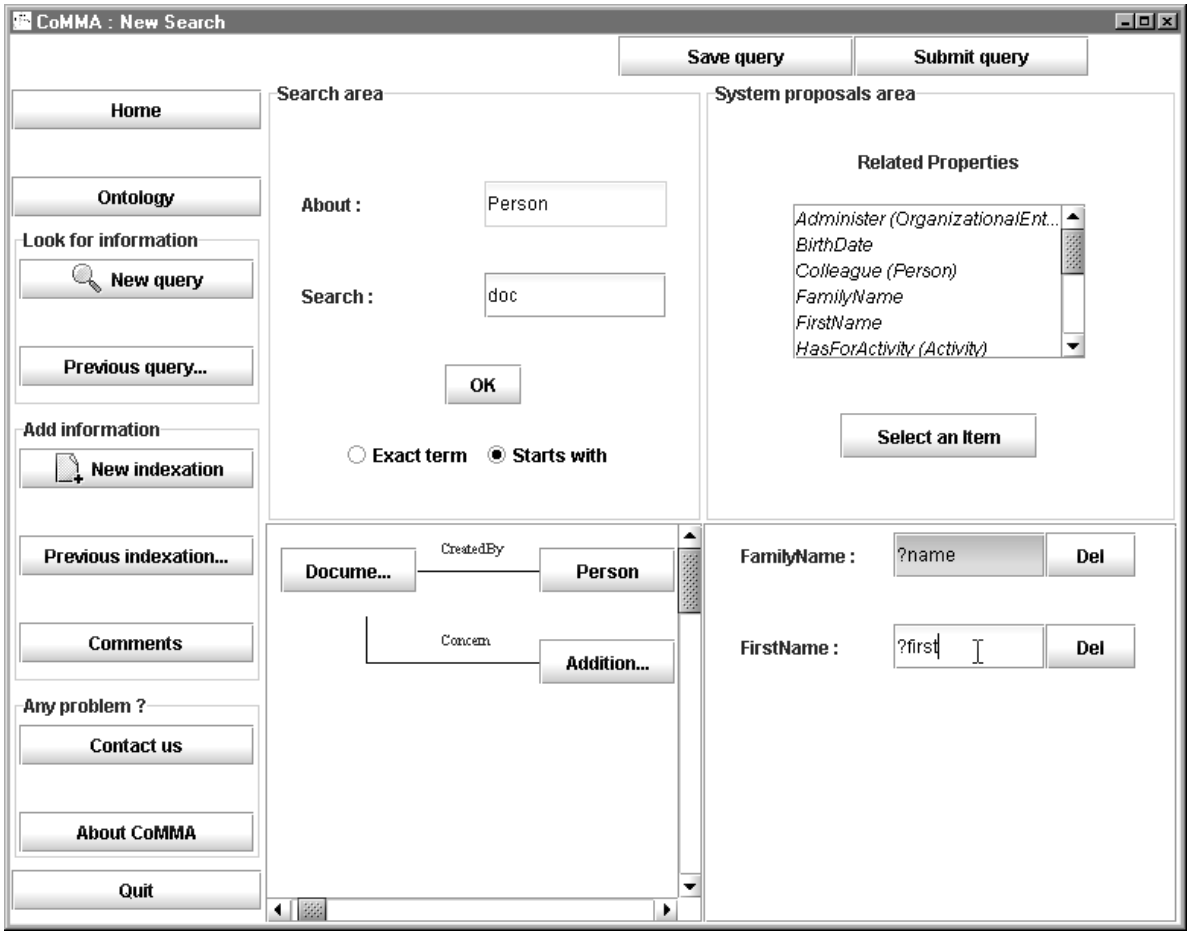


Figure 163 Interface de requêtes

Après l'identification des rôles vient la spécification des interactions entre rôles. Le modèle de chaque conversation doit être indiqué avec son protocole. Les agents doivent se conformer aux protocoles afin que le SMA fonctionne correctement. La définition d'un protocole commence par un graphe d'accointances au niveau des rôles, celui-ci est un graphe orienté identifiant des canaux de communication entre les agents jouant les rôles considérés. Nous indiquons alors l'ordre possible des messages échangés. Le réseau d'accointances et les protocoles dérivent de l'analyse architecturale et des cas d'utilisation isolés à partir des scénarios d'application.

Le protocoles utilisent des primitives (ex: inform, cfp, accept, reject...) qui dénotent des actes du langage spécifiés et normalisés dans une ontologie recommandée par le consortium FIPA. Tout agent utilisant et suivant ces protocoles est en droit d'attendre un comportement 'compréhensible' de la part de ses interlocuteurs.

Plusieurs ontologies sont utilisées pour la rédaction d'un message. Un message envoyé par un agent est décomposable en trois niveaux:
- le premier niveau utilise des primitives standardisées par l'ontologie de FIPA pour décrire la nature d'un message entre deux agents.
- le deuxième niveau utilise des primitives consensuelles entre les programmeurs du prototype de CoMMA ; elles sont capturées par une ontologie gérée par la plate-forme JADE et fournissent un vocabulaire spécialisé pour les actes du langage utilisés dans la gestion d'un intraweb sémantique.
- le troisième niveau utilise des primitives de O'CoMMA pour décrire les annotations recherchées.

En partant des descriptions des rôles et des interactions, les différents membres du consortium CoMMA ont proposé et implanté des types d'agent qui remplissent un ou plusieurs rôles impliqués dans la gestion et l'exploitation de la mémoire.

L'aspect faiblement couplé des agents et le consensus ontologique supportant un couplage au niveau sémantique ont permis de répartir le développement tout en assurant une bonne facilité d'intégration pour le prototype final. Les comportements des agents proviennent des choix d'implantation déterminant les réponses, les actions et les réactions de l'agent. L'implantation d'un comportement d'agent est contrainte par le rôle qu'il joue et est sujette à la boîte à outils des techniques disponibles aux concepteurs.

Le faible couplage a permis à chacun des partenaires d'intégrer dans ses agents des techniques de son domaine d'expertise (ex: apprentissage symbolique, etc.) tout en permettant au système intégré de bénéficier dans son ensemble de l'association de ces techniques pour un objectif commun: gérer et diffuser la connaissance manipulée. Chaque partenaire est tenu de programmer le comportement de son agent afin qu'il réponde à tous les messages qui peuvent lui être envoyer, sachant que ceux-ci utiliseront toujours les primitives de l'ontologie qui est disponible pour tout agent par simple demande auprès de l'agent ontologiste.

14.7 Cas de la société dédiée aux annotations

L'enjeu de cette société est de trouver des mécanismes pour décider où stocker les annotations nouvellement soumises et comment distribuer une requête pour ne pas laisser échapper des réponses simplement parce que l'information nécessaire est dispersée entre plusieurs bases d'annotation. Ces deux facettes de la distribution sont liées puisque le processus de résolution distribuée d'une requête dépend étroitement du choix fait pour la distribution des annotations. Dans cette société le médiateur d'annotation (AM) est responsable de la gestion de la dispersions des annotations ; ces annotations étant réparties entre des archivistes d'annotation (AAs).

Pour allouer une annotation nouvellement soumise, l'AM émet un appel à proposition à l'attention des AAs. Un protocole d'interaction correspondant à un 'contract-net' [Davis and Smith, 1983] imbriqué est utilisé pour déterminer lequel des AAs gagnera l'annotation nouvellement soumise. En réponse, chaque AA mesure la proximité sémantique entre l'annotation et les types des concepts et de relations actuellement dans ses archives. L'AA le plus proche gagne le contrat. Ce protocole permet de spécialiser les bases d'annotations sémantiques de la mémoire et d'entretenir cette spécialisation. Pour ce protocole, nous avons donc défini une pseudo-distance en utilisant la hiérarchie de l'ontologie et une distance lexicographique pour les valeurs littérales. La distance utilisant la hiérarchie est basée sur le chemin entre deux notions, passant par le plus proche commun supertype ; les supertypes communs représentent ce que deux notions ont en commun. Nous l'employons pour comparer les offres des différents AAs. L'ontologie est ainsi utilisée comme un espace commun permettant de définir des (pseudo-)distances communes dont les résultats sont comparables. Le consensus ontologique fournit donc une base pour d'autres consensus (ex: un consensus sur le calcul d'une métrique partagée). L'objet 'ontologie' est donc la pierre de touche de tous les mécanismes distribués d'une gestion intelligente des connaissances dispersées.

La résolution d'une requête peut impliquer plusieurs bases d'annotations réparties entre plusieurs AAs ; le résultat est une fusion de résultats partiels. Pour déterminer si et quand un AA doit participer à la résolution d'une requête, les AAs calculent le recouvrement entre la liste des notions actuellement utilisées dans leur base et la liste de celles utilisées dans la requête (en prenant en compte les liens de subsomption dans l'ontologie). En utilisant ces descriptions de recouvrement, l'AM peut identifier à chaque étape de son algorithme de décomposition et pour chaque requête intermédiaire qu'il produit, les AAs à consulter. L'ontologie partagée fournit ici les primitives permettant de décrire des connaissances allouées à chaque agent et permet ainsi de statuer sur la pertinence d'une participation d'un agent à une tâche donnée.

Une fois que les rôles d'AA et d'AM ont été spécifiés avec leurs interactions, des modules de CORESE ont été intégrés dans les types d'agent implantant ces rôles afin de leur fournir les compétences nécessaires.

J'ai détaillé l'exemple de cette société pour montrer comment les techniques de l'ingénierie des connaissances pouvaient fournir les outils nécessaires pour que d'autres domaines puissent apporter leur contribution à la résolution de problèmes complexes. Ainsi, l'approche reposant sur la mise en place d'une ontologie partagée permet à un SMA de participer à la gestion de connaissances distribuées. Inversement, le projet FRODO [Van Elst and Abecker, 2001] montre comment un SMA peut assister la mise en place et le maintien d'un consensus ontologique à l'intérieur d'une communauté et entre des communautés.

14.8 Discussion et perspectives

Dans ce résumé, j'ai présenté une vision de la mémoire d'entreprise ainsi que l'approche évaluée dans le projet CoMMA basée sur les systèmes multi-agents, l'apprentissage symbolique et le Web sémantique. J'ai montré que l'ontologie donne la clef de voûte d'un tel système d'information multi-agents et j'ai détaillé chaque étape de sa conception.

L'ontologie est un puissant objet conceptuel pour la modélisation. Sa complexité m'a amené à développer des méthodologies et des outils d'aide à sa conception qui doivent tendre vers des plates-formes intégrées supportant le cycle de vie complet de ces objets vivants. Cependant cette complexité se manifeste aussi par une augmentation de la complexité des solutions dans lesquelles les ontologies sont introduites. En particulier le fossé entre les modèles conceptuels sous-jacents à un système et les préoccupations quotidiennes d'un utilisateur lambda de cette solution se creuse dangereusement.

J'ai présenté une approche innovatrice pour la gestion d'une mémoire organisationnelle combinant l'ingénierie d'ontologie, le Web sémantique et les systèmes multi-agents dans une solution intégrée. CoMMA appartient donc à la famille des SIMA qui s'intéressent à la gestion des connaissances dans une entreprise et se concentre sur la gestion d'une mémoire de documents hétérogènes mais avec des annotations homogènes et basées sur une ontologie partagée en reposant sur les technologies du Web sémantique. Les aspects que j'ai plus particulièrement étudiés sont la recherche, la suggestion et l'archivage des annotations dans un environnement distribué. La résolution d'une requête ou l'allocation d'une nouvelle annotation reposent sur des interactions entre agents dont les protocoles exploitent les primitives définies par l'ontologie O'CoMMA. Le consensus ontologique est bien la clef de voûte des mécanismes de gestion des connaissances distribuées.

Le prototype implanté en JAVA a été évalué par des utilisateurs d'une compagnie de télécommunication (T-System Nova Deutsch Telekom) et d'un organisme de recherche sur le Bâtiment (CSTB) avec des archives contenant jusqu'à 1000 annotations.

Des problèmes d'interface et d'ergonomie ont été soulevés par les utilisateurs, mais l'utilité et le potentiel des fonctionnalités offertes par le système ont été unanimement reconnus. En particulier, les apports d'une ontologie et des techniques basées sur les langages de modélisation des connaissances ont été très appréciés par les utilisateurs finaux. De même, les développeurs du prototype ont vu dans le couple agent-ontologie une nouvelle façon d'aborder la spécification et la répartition d'un travail d'implantation.

J'ai montré ici que l'intégration d'une approche d'ingénierie des connaissances (formalisation de connaissances sur les ressources dispersées dans un intraweb par des annotations sémantiques basées sur des ontologies) et de l'intelligence artificielle distribuée (systèmes d'information de plusieurs agents couplés par une coopération au niveau sémantique) peut constituer un nouveau paradigme puissant pour la résolution d'une classe de problèmes distribués complexes. Il reste, évidemment, d'autres points à approfondir. En premier lieu, dans le cadre du cycle de vie de la mémoire d'entreprise, il reste à étudier pour chaque étape l'intérêt d'une telle approche.

Enfin, les organisations ne sont pas des îlots isolés de la société mais participent elles-mêmes à des réseaux d'interactions complexes. Il faudra donc explorer l'extension de solutions similaires aux contextes d'extranet, voire, pour la communauté du Web sémantique, au Web ouvert.

15 O'CoMMA

15.1 Lexicon view of concepts

Terms	Concept ID	Parent IDs	Natural Language definition
abstract,	Abstract	Document	Document corresponding to a summary of another document.
academic,	Academic	Education Research	Activity of education and/or research usually affiliated to academic institution.
acoustics,	AcousticsBuildingTopic	ScienceAppliedToBuildingTopic	12
activity,	Activity	NonSpatialEntity	Entity representing a type of voluntary action.
activity able entity,	ActivityAbleEntity	RoleEntity	Entity that can have an activity.
addition,	Addition	CorporateMemoryEvent	Corporate Memory Event corresponding to content added to the memory.
additional topic,	AdditionalTopic	Entity	Entity representing subjects that hold attention and possibly something one wants to discover. These topics are additional in the sens that they were not introduced or used anywhere else in the ontology but were identified as relevant for document annotation - domain concepts, general subjects...-
administration, administer,	Administration	Service	Service of supervision and control of the operations or arrangement of an Organizational Entity.
administration able entity,	AdministrationAbleEntity	RoleEntity	Entity that can administrate another.
administrator,	Administrator	Professional	Professional who is responsible for managing its organizational affairs. Administrator may or may not also be required to manage people. If so, then they are also Managers.
advertise, advertising, advertisement,	Advertise	Service	Activity consisting in making something known in order to sell it.
advertisement, publicity, promotion,	Advertisement	Document	Document of a public announcement, especially to proclaim the qualities or advantages of some product or service so as to increase sales.
aerodynamics,	AerodynamicsBuildingTopic	ScienceAppliedToBuildingTopic	13
agriculture,	AgricultureTopic	AdditionalTopic	Science of cultivating the soil, producing crops, and raising livestock; farming.
AIFF , A.I.F.F., Audio Interchange File	AIFFFormat	DataFileFormat	Data File Format of audio documents encoded in Audio Interchange File Format.

Format,			
air pipe,	AirPipeTopic	PipeAndShaftTopic	235
algebra,	AlgebraTopic	MathematicsTopic	the mathematics of generalized arithmetical operations.
American,	American	English	English spoken by the natives of American.
analog format,	AnalogFormat	StorageFormat	Storage format based on continuous data.
analysis,	AnalysisTopic	MathematicsTopic	Branch of mathematics involving calculus and the theory of limits, sequences, series, integration and differentiation.
animation,	AnimationRepresentation	DynamicImageRepresentation	Dynamic Image that appears to move.
annual report,	AnnualReport	Report	Report of the formal financial statements and operations, issued by a corporation to its shareholders after its fiscal year-end.
aquaculture,	AquacultureTopic	AdditionalTopic	Science, of cultivating marine or freshwater food fish or shellfish, such as oysters, clams, salmon, and trout, under controlled conditions.
architecture,	ArchitectureTopic	BuildingTransversalTopic	63
archivist,	Archivist	TechnologyMonitoringActor	Technical monitoring actor responsible for library activities.
area referent,	AreaReferent	TechnologyMonitoringActor	Technical monitoring actor responsible for an expertise area and who has a group of contributors observers to manage.
art,	Art	Activity	Activity through which people express particular ideas, the making of what is expressive or beautiful.
article,	Article	Document ExtractedDocument	Document corresponding to a piece of writing on a particular subject and which purpose is to fully realize a particular objective in a relatively concise form e.g.: demonstrate something.
artificial intelligence, AI, A.I.,	ArtificialIntelligenceTopic	ComputerScienceTopic	Branch of computer science concerned with making computers behave like humans.
artificial language,	ArtificialLanguage	Language	Language that has been explicitly developed at a specific time, rather than evolving naturally over time through use by a community. It is usually designed for a specific purpose.
asphalt,	AsphaltTopic	BuildingMaterialTopic	25
assembly language,	AssemblyLanguage	LowLevelLanguage	Low-level Language that contains the same instructions as a Machine Language, but the instructions and variables have names instead of being just numbers.
assistant,	Assistant	Professional	Professional that contributes to the fulfillment of a need or furtherance of an effort or purpose.
association,	Association	GroupOfIndividuals	Formal group of individuals who have an interest, an activity, or a purpose in common.
attribute,	Attribute	NonSpatialEntity	Predefined entities characterizing other entities.
audio CD, audio compact disc,	AudioCD	CD	CD where the Documentary Element uses Audio Perception and is recorded using hi-fi music industry format.
auditor, inspector,	AuditorInspector	Professional	Professional performs audit.
auditory,	AudioPerceptionMode	DocumentaryPerceptionMode	Documentary Perception Mode by hearing.
bad,	BadRating	RatingValue	Value corresponding to a bad feedback given about a consultation
banking,	Banking	Finance	Finance activity concerned with deposits, channeling money into lending activities, providing services for investing money, borrowing money or changing money to foreign currency.
benevolent association,	BenevolentAssociation	Association	Association which helps a particular group of people in need.
biology,	BiologyTopic	AdditionalTopic	The scientific study of the natural processes of living things.
book,	Book	Document	Document consisting of a set of pages to be fastened together inside a cover to be read.
booklet, leaflet, pamphlet,	Booklet	Book	Book (small) with a small number of pages often giving information about something.
brick,	BrickTopic	EarthenwareTopic	25
building,	BuildingTopic	AdditionalTopic	
building materials,	BuildingMaterialTopic	MaterialAndWorkTechnicsTopic	25
building product,	BuildingProductTopic	MaterialAndWorkTechnicsTopic	25
building site,	BuildingSiteTopic	MaterialAndWorkTechnicsTopic	27
building transversal theme,	BuildingTransversalTopic	BuildingTopic	60
business,	Business	Activity	Activity composed of commercial or industrial activities intended to make profits.
buy, purchase,	Buy	Commerce	Commerce activity where a person or an organization obtains something by paying money for it.
cartography,	CartographyTopic	EarthObservationTopic	
catalog, catalogue,	Catalog	ReferenceDocument	Reference document containing an enumeration of things usually linked to a domain or an activity.
CD, C.D., compact disc,	CD	DocumentaryMedium	Digital Medium where the Documentary Element is recorded on a optical disc.
CD audio track,	CDAudioTrack	DigitalFormat	Digital format used for audio track on compact discs.
CD-ROM,	CD-ROM	CD	CD used for storing computer data.
cement,	CementRenderingTopic	CoatingTopic	25
chart,	Chart	Document	Document corresponding to a visual display of information often intended to show the information more clearly.
civil engineering,	CivilEngineeringBuildingTopic	ScienceAppliedToBuildingTopic	17

climatology,	ClimatologyBuildingTopic	ScienceAppliedToBuildingTopic	18
clock time, time,	ClockTime	TimeEntity	Time Entity corresponding to a time of day.
club,	Club	GroupOfIndividuals	Group of individuals with a common purpose or interest who meet regularly and take part in shared activities.
cluster,	Cluster	OrganizationPart	Organization part grouping projects according to their client type.
coating,	CoatingWorkTopic	SecondWorkTopic	233
coating,	CoatingTopic	BuildingMaterialTopic	25
code of conduct,	CodeOfConduct	Manual	Manual that makes explicit the expectations governing the behavior of those agents subject to it in certain kinds of situations.
cognitive sciences,	CognitiveSciencesTopic	AdditionalTopic	Topic concerned with the sciences studying cognition.
commerce,	Commerce	Service	Activity concerned with buying or selling goods or services for a profit.
competitor,	Competitor	Organization	Organization having similar activities to an organization and trageting the same markets.
computer algebra methods,	ComputerAlgebraMethodsTopic	ProgrammingTopic AlgebraTopic	
computer analysis,	ComputerAnalysisTopic	ArtificialIntelligenceTopic	
computer file format,	ComputerFileFormat	DigitalFormat	Digital format corresponding to a single collection of computer data or information that has a name.
computer graphics,	ComputerGraphicsTopic	ComputerScienceTopic	
computer science,	ComputerScienceTopic	AdditionalTopic	Study of automatic information and data processing methods, and of computers, including both hardware and software design.
computer vision,	ComputerVisionTopic	ImageProcessingTopic	
concrete,	ConcreteTopic	BuildingMaterialTopic	25
conference, lecture,	Conference	FormalGathering	Formal Gathering of persons with common interests, esp. professional interests, for the purpose of sharing information and opinions especially through lectures and debates.
confidential document,	ConfidentialDocument	DiffusionRight	The diffusion is restricted to a defined community. There are different levels of confidentiality for internal or external diffusion. A document can be composed of confidential parts or not confidential parts (for example: the reference and the abstract can be freely diffused but not the integral text).
consortium,	Consortium	Organization	Organization of several Businesses joining together as a group for some shared definite purpose.
consultancy report,	ConsultancyReport	Report	Report on studies performed for clients. Most of them are confidential.
consultant,	Consultant	Professional	Professional who works with some business in a consulting capacity.
consultation,	Consultation	CorporateMemoryEvent	Corporate Memory Event corresponding to a user seeking information from the memory.
consultation trace,	ConsultationTrace	NonSpatialEntity	Such an element is created when the user visits a document, and is updated every time he returns to the document.
consulting,	Consulting	Service	Service of providing professional advice or expertise.
contract type attribute,	ContractTypeAttribute	Attribute	Type of contract.
CORBA, C.O.R.B.A., common object request broker architecture,	CORBATopic	NetworkTopic ObjectProgrammingTopic	Common Object Request Broker Architecture, an architecture that enables pieces of programs, called objects, to communicate with one another regardless of what programming language they were written in or what operating system they are running on.
corporate memory,	CorporateMemoryTopic	KnowledgeEngineeringTopic	An explicit, disembodied and persistent representation of knowledge and information in an organization, in order to facilitate their access and reuse by members of the organization, for their tasks.
corporate memory event,	CorporateMemoryEvent	Event	Event corresponding to changes in the Corporate Memory.
corrosion,	CorrosionTopic	PathologyTopic	26
course, training document,	CourseDocument	Document	Training document containing pedagogical material and usually used by lecturers or teachers as a support of their lessons.
customer, client,	Customer	Organization	Organization who pays for goods or services.
data file format,	DataFileFormat	ComputerFileFormat	Computer File format dedicated and formatted to be manipulated by a Program.
date,	Date	TimeEntity	Time Entity corresponding a numbered day in a month, often given with a combination of the name of the day, the month and the year.
deletion,	Deletion	CorporateMemoryEvent	Corporate Memory Event corresponding to content removed from the memory.
department,	Department	OrganizationPart	Thematic grouping of services.
development,	Development	Service	Service corresponding to analysis, design, implementation and testing of research solutions.
diagram,	Diagram	Chart	Chart intended to explain how something works a drawing showing the relation between the parts.
dictionary,	Standard	ReferenceDocument NumberableEntity	Reference used as a point of reference to compare and evaluate quality of products or systems.
dictionary,	Dictionary	ReferenceDocument	Reference Document in which words are listed alphabetically and their meanings.
diffusion right,	DiffusionRight	DocumentAttribute	Authorized scope for the diffusion of a document.

digital format, numeric format,	DigitalFormat	StorageFormat	Storage format based on discontinuous data.
direction,	Direction	OrganizationPart	Organization part with a special activity inside the company eg: HR, Project Planning.
discrete event simulation,	DiscreteEventSimulationTopic	SimulationTopic	
distributed artificial intelligence, DAI, D.A.I.,	DistributedArtificialIntelligenceTopic	ArtificialIntelligenceTopic	Distributed approach of Artificial Intelligence.
division,	Division	OrganizationPart GroupOfIndividuals	Functional subdivision of a service. It has generally 10 to 25 people.
document,	Document	Entity EntityConcerningATopic NumberableEntity	Entity including elements serving as a representation of thinking.
documentary file,	DocumentaryFile	Document	Thematic document, regularly updated and made with heterogeneous material (articles, references, synthesis…).
documentary medium,	DocumentaryMedium	PhysicalEntity	Physical entity through which signals/messages travel as a means for communication.
document attribute,	DocumentAttribute	Attribute	Attribut characteristic of documents.
document orign,	DocumentOrign	DocumentAttribute	Where the document was produced or written.
door,	DoorTopic	JoineryTopic	231
drainage,	DrainageTopic	OutsideDevelopmentTopic	21
duration,	Duration	TimeEntity	Time Entity corresponding to the length of time that something lasts.
DVD, D.V.D., digital versatile disc, digital video disc,	DVD	DocumentaryMedium	Digital Medium where the Documentary Element is recorded on a optical disc called Digital Versatile Disc or Digital Video Disc.
DVD-ROM,	DVD-ROM	DVD	DVD used for storing computer data.
dynamic image,	DynamicImageRepresentation	ImageRepresentation	Image changing over time quickly enough to be noticed.
dynamic systems,	DynamicSystemsTopic	MathematicsTopic	
earthenware, terracotta,	EarthenwareTopic	BuildingMaterialTopic	25
earth observation,	EarthObservationTopic	AdditionalTopic	
economic science,	EconomicScienceTopic	AdditionalTopic	30
education,	Education	Service	Service of teaching and/or training.
e-mail, electronic mail, mail,	E-Mail	Mail	Mail sent in electronic format over a computerized world-wide communication system.
employee,	Employee	Professional	Professional who works for an organization in return for financial or other compensation. Disjoint with Self-employed Professional.
employee manual,	EmployeeManual	Manual	Manual that officially explains company policies, procedures, and benefits.
encyclopedia, encyclopaedia,	Encyclopedia	ReferenceDocument	Reference Document usually rather large and containing many articles arranged in alphabetical order which deal either with the whole of human knowledge or with a particular part of it.
energetic engineering,	EnergeticEngineeringTopic	EquipmentTopic	241
energy,	EnergyBuildingTopic	ScienceAppliedToBuildingTopic	19
engineer,	Engineer	Professional	Professional who works in some branch of engineering using scientific knowledge to solve practical problems.
English,	English	NaturalLanguage	Natural Language spoken by the natives of England.
entertainment event,	EntertainmentEvent	Event	Event occurring primarily to amuse or entertain Persons.
entity, thing,	Entity	Something	Thing which exists apart from other Things, having its own independent existence and that can be involved in Events.
entity concerning a topic,	EntityConcerningATopic	RoleEntity	Entity that can concern a topic.
environment,	EnvironmentTopic	BuildingTransversalTopic	64
equipment, facilities, installations,	EquipmentTopic	MaterialAndWorkTechnicsTopic	24
event,	Event	Something	Thing taking place, happening, occurring and usually recognized as important, significant or unusual.
Excel, Excel format, Microsoft Excel format,	MSExcelFormat	DataFileFormat	Data File Format of a visual document encoded in the format of Microsoft Excel workbook.
executable format, EXE,	ExecutableFormat	ComputerFileFormat	Computer File Format of a program that can be executed by a computer.
executive,	Executive	Professional	Professional who holds a high position in some Organization, makes decisions and puts them into action.
external document,	ExternalDocument	DiffusionRight	Document acquired from an external source and not written in-house.
external people,	ExternalPeople	Professional	Professional from an organization but working for another for a limited period of time.
extracted document,	ExtractedDocument	Document	Document that can have been extracted from another.

facing, wall coating,	FacingTopic	CoatingWorkTopic	233
female,	Female	SexualAttribute	Person of the sex that can give birth.
ferrous metal,	FerrousMetalTopic	MetalTopic	25
filtering,	FilteringTopic	ProbabilitiesTopic StatisticsTopic	
final report,	FinalReport	Report	Report concluding and synthesizing the results of a research action or a consultancy contract.
finance,	Finance	Service	Service concerned with buying, selling, trading, converting, or lending money, in the form of currency or negotiable financial instruments -such as stocks, bonds, commodities futures, etc.-
flooring, floor coating,	FlooringTopic	CoatingWorkTopic	233
flowchart,	Flowchart	Diagram	Diagram which shows the stages of a process.
form,	Form	Document	Document for structured solicitation of input from a user.
formal gathering,	FormalGathering	Gathering	Gathering agreeable to established mode, forms, conventions and requirements, methodical well planned and organized, not incidental, sudden or irregular.
foundations,	FoundationsTopic	ShellTopic	225
French,	French	NaturalLanguage	Natural Language spoken by the natives of France.
front, facade,	FacadeBuildingTopic	ShellTopic	224
full time,	FullTime	ContractTypeAttribute	Contract type of an employee working full-time for an organization.
functional equations,	FunctionalEquationsTopic	MathematicsTopic	
gathering,	Gathering	Event SituableEntity EntityConcerningATopic	Event corresponding to the social act of a group of Persons assembling in one place.
gathering entity,	GatheringEntity	RoleEntity	Entity that can participate to a gathering.
geometry,	GeometryTopic	MathematicsTopic	Branch of mathematics relating to the study of space and the relationships between points, lines, curves and surfaces.
German,	German	NaturalLanguage	Natural Language spoken by the natives of Germany.
GIF, G.I.F., graphics interchange format,	GIFFormat	DataFileFormat	Data File Format of a visual document compressed in Graphics Interchange Format.
glass,	GlassMaterialTopic	BuildingMaterialTopic	25
good,	GoodRating	RatingValue	Value corresponding to a good feedback given about a consultation.
GPRS, G.P.R.S., general packet radio service,	GPRSTopic	MobilePhoneProtocolsTopic	
graph,	Graph	Chart	Chart which shows a series of points, lines, line segments, curves, or areas that represents the variation of a variable in comparison with that of one or more other variables.
graphic,	GraphicRepresentation	IconicRepresentationSystem	Iconic Representation written or drawn, printed or engraved.
groupable entity,	GroupAbleEntity	RoleEntity	Entity that can be included in an other entity (a group).
group of individuals,	GroupOfIndividuals	OrganizationGroup	Organization Group composed of individuals only.
group profile,	GroupProfile	Profile	Profile concerning a group of people.
GSM, G.S.M., global system for mobile communications,	GSMTopic	MobilePhoneProtocolsTopic	
HCI, H.C.I., human-computer interaction,	HCITopic	ComputerScienceTopic	Study of human-computer relations with the aim to improve them.
head,	Head	Professional	Professional responsible for an organization part.
head of department,	HeadOfDepartment	Head	Professional responsible for a department.
head of division,	HeadOfDivision	Head	Professional responsible for a division.
head of laboratory, laboratory manager,	HeadOfTestLaboratory	Head	Professional responsible for a laboratory.
head of pole,	HeadOfPole	Head	Professional responsible for a pole.
head of project,	HeadOfProject	Head	Professional responsible for a project.
head of service,	HeadOfService	Head	Professional responsible for a service.
high level language,	HighLevelLanguage	ProgrammingLanguage	Programming Language that enables a programmer to write programs that are more or less independent of a particular type of computer. Such languages are considered high-level because they are closer to human languages than are Low-level Languages, which are closer to machine languages.
home page, welcome page, homepage,	Homepage	WebPage	Web Page designed to be the main page of a Web site. Typically, the home page serves as an index or table of contents to other documents stored at the site.
house automation,	HomeAutomationTopic	EquipmentTopic	244
HTML, H.T.M.L.,	HTMLFormat	TextFormat	Text Format of a Documentary Element written in HyperText Markup Language.

H.T.M.L. format, HyperText Markup Language format,			
HTML, H.T.M.L., hypertext markup language,	HTMLTopic	WebTopic	Web language for hypertext markups.
HTTP, H.T.T.P., hypertext transfer protocol,	HTTPTopic	WebTopic	Web protocol for transferring hypertext.
human resources,	HumanResources	Service	Service of administration of people, especially the skills and abilities they have, and the job and position they occupy.
human science,	HumanScienceTopic	AdditionalTopic	30
hygrothermics,	HygrothermicsBuildingTopic	ScienceAppliedToBuildingTopic	15
iconic representation system,	IconicRepresentationSystem	DocumentaryRepresentationSystem	Documentary Representation System based on icons that perceptually resemble the objects they represent - e.g. the drawing of an apple and a real apple -
illustration,	Illustration	Document	Document corresponding to artworks that help make something clear or attractive.
image, picture,	ImageRepresentation	GraphicRepresentation	Graphic visually representing an object, a scene, a person... and produced on a surface.
image processing,	ImageProcessingTopic	ComputerGraphicsTopic	Analyzing and manipulating images with a computer.
independent contractor,	IndependentContractor	SelfEmployed	Contract type of a self-employed professional who is retained to perform a certain act but who is subject to the control and direction of another only as to the end result and not as to the way in which the act is performed.
index,	Index	Document	Document consisting of summary list of other items.
index card,	IndexCard	Document	Document for categorising services, applications, company s market operations, etc.. This is a useful way to represent information when the emphasis is put on some specific point of innovation rather than on a very detailed description of the single service, application or fact.
indexical representation system,	IndexicalRepresentationSystem	DocumentaryRepresentationSystem	Representation System based on indices that are physically connected with the object they represents - e.g. smoke and fire.
individual profile,	IndividualProfile	Profile	Profile concerning one individual.
inert entity, lifeless entity,	LifelessEntity	PhysicalEntity	Physical Entity that cannot be alive.
informal gathering,	InformalGathering	Gathering	Gathering without official forms - of clothing, behavior, speech - not according to conventional, prescribed, or customary forms or rules hence, without ceremony not officially recognized or controlled, usually having or fostering a warm or friendly atmosphere, especially through smallness.
information form,	InformationForm	Form	Form for engineers and researchers to share their (informal) information.
insulation,	InsulationTopic	SecondWorkTopic	232
insulator,	InsulatorTopic	BuildingMaterialTopic	25
insurance,	Insurance	Service	Service of providing financial and material protection to clients in the event of sickness, death, natural disaster, loss, theft, lawsuits, etc.
integration process actor,	IntegrationProcessActor	Person	A Person playing a role in the newcomer integration process.
interactivity,	InteractivityTopic	HCITopic	Study of sensory dialog that occurs between a human being and a computer system.
interest able entity,	InterestAbleEntity	RoleEntity	Entity that can show interests in some topics.
intermediate report,	IntermediateReport	Report	Punctual report produced at the end of each step of a research action or a consultancy contract (state of the art, experiment…).
internal document,	InternalDocument	DiffusionRight	The document was produced in-house..
international organization, multinational,	InternationalOrganizationGroup	OrganizationGroup	Organization Group of international scope, that is, one which has substantial operations, physical facilities, or substantial membership in multiple countries.
Internet,	InternetTopic	NetworkTopic	A global network connecting millions of computers.
intern mail,	InternMail	Mail	Mail transmitted via an Organization internal post system.
interview,	Interview	Meeting	Meeting face to face to ask a series of questions usually in order to obtain information from the interviewee. There is usually one interviewee - person who is asked questions - and one interviewer - person who asks the questions but there may be more.
ISSN holder document,	ISSNHolderDocument	Document	Document that can have an ISSN.
Italian,	Italian	NaturalLanguage	Natural Language spoken by the natives of Italy.
java programming,	JavaProgrammingTopic	ObjectProgrammingTopic	Progamming in JAVA object-oriented language.
joinery, carpentry,	JoineryTopic	SecondWorkTopic	231
journal,	Journal	ISSNHolderDocument	Document corresponding to a serious magazine or newspaper which is published regularly, usually about a specialist subject.
JPEG, J.P.E.G., J.P.E.G. format, joint photographic experts group	JPEGFormat	DataFileFormat	Data File Format of a visual document compressed in Joint Photographic Experts Group format.

format,			
knowledge acquisition,	KnowledgeAcquisitionTopic	KnowledgeEngineeringTopic	Field dealing with techniques for acquiring knowledge.
knowledge based systems,	KnowledgeBasedSystemsTopic	KnowledgeEngineeringTopic	
knowledge dissemination,	KnowledgeDissemination	Service	Diffusing knowledge previously acquired.
knowledge engineering,	KnowledgeEngineeringTopic	CognitiveSciencesTopic ArtificialIntelligenceTopic	Field dealing with knowledge acquisition, representation, validation, inferencing, explanation and maintenance.
knowledge management,	KnowledgeManagementTopic	KnowledgeEngineeringTopic	Field dealing with management techniques for knowledge capitalization in an organization.
knowledge modeling,	KnowledgeModelingTopic	KnowledgeEngineeringTopic	Field dealing with modeling techniques for representing knowledge.
language, tongue,	Language	SymbolicRepresentationSystem	Symbolic Representation System for communication consisting of a set of small parts and a set of rules which decide the ways in which these parts can be combined to produce messages that have meaning.
lecture, course, talk,	Lecture	FormalGathering	Formal Gathering where a Person - lecturer - teaches by giving a discourse on some subject to a group of people students-.
legal,	Legal	Service	Activity concerned by the respect of the law.
legal corporation,	LegalCorporation	Organization	Organization which is a private, legal, corporate entity with the legal rights to own property, able tomanage itself, and sue or be sued. It is established by a charter or registration granted by a government.
letter,	Letter	PostMail	Post Mail written or printed, usually put in an envelope and respecting a standard presentation.
lexicon,	Lexicon	Dictionary	Dictionary usually small and limited to a particular language or subject.
lighting,	LightingBuildingTopic	ScienceAppliedToBuildingTopic	15
lighting,	LightingTopic	EquipmentTopic	243
liquid mechanics,	LiquidMechanicsTopic	MechanicsTopic	
living being, living entity,	LivingEntity	PhysicalEntity	Physical Entity that can be alive.
local organization, regional organization, local organisation, regional organisation,	LocalOrganizationGroup	OrganizationGroup	Organization Group of local scope, that is, members distributed in a local area - a Neighborhood, City, rural region, etc. - or having a local area of activity and concern.
location,	Location	SpatialEntity NumberableEntity	Spatial Entity which is a point or an extent in space.
logo,	Logo	Document	Document showing the emblem of a group.
low level language,	LowLevelLanguage	ProgrammingLanguage	Programming Language that is closer to the hardware than are High-level Languages, which are closer to human languages.
machine language,	MachineLanguage	LowLevelLanguage	The lowest-level Programming Language that consistw entirely of numbers.
magazine,	Magazine	ISSNHolderDocument	Document corresponding to a type of thin book with large pages which contains articles and photographs. It is usually intended to be a weekly or monthly paperback publication.
mail,	Mail	Document	Documents sent and delivered through a dedicated conveyance network.
make something,	MakeSomething	Activity	Activity in which something - tangible - is made from some raw materials.
male,	Male	SexualAttribute	Person who belongs to the sex that cannot give birth.
manageable entity,	ManageableEntity	RoleEntity	Entity that can be managed.
management able entity,	ManagementAbleEntity	RoleEntity	Entity that can manage another.
management board,	ManagementBoard	OrganizationPart GroupOfIndividuals	Sub group of an organization formed by the heads of services.
management committee,	ManagementCommittee	OrganizationPart GroupOfIndividuals	Sub group of an organization formed by the President, the Director, the Director of Research and Development and the Technical Director.
manager, chief,	Manager	Professional ManagementAbleEntity	Professional whose primary job is to manage other people, directing their work activity. A Manager tells his or her subordinate workers what to do.
manual, instructions,	Manual	ReferenceDocument	Reference Document which gives you practical instructions on how to do something or how to use something, such as a machine.
manufacture,	Manufacturing	MakeSomething	Make Something from raw materials or component parts that are combined to produce a product.
map,	Map	Document	Document which, properly interpreted, models a region of physical space many times its own size by using graphical symbols - or possibly another code -, often in conjunction with a natural language.
marketing,	Marketing	Service	Service by which a product or service is sold, including assessment of its sales potential and responsibility for its promotion and distribution.
markovian models,	MarkovianModelsTopic	MathematicalModellingTopic	
material and work technics,	MaterialAndWorkTechnicsTopic	BuildingTopic	20
mathematical modelling,	MathematicalModellingTopic	MathematicsTopic	
mathematics,	MathematicsTopic	AdditionalTopic	Topic concerned with the study of numbers, shapes and space using reason and usually a special system of symbols and rules for organizing them.

measurement,	MeasurementBuildingTopic	ScienceAppliedToBuildingTopic	11
mechanical resistance,	MechanicalResistanceBuildingTopic	ScienceAppliedToBuildingTopic	14
mechanics,	MechanicsTopic	PhysicsTopic	Study of the effect of physical forces on objects and their movement.
medical care,	MedicalCare	Activity	Activity of medical care of patients, including surgery, psychological care, physical therapy, practical nursing, and dispensing drugs.
medical images processing,	MedicalImagesProcessingTopic	ImageProcessingTopic	
medium,	MediumRating	RatingValue	Value corresponding to a medium feedback given about a consultation
meeting,	Meeting	FormalGathering	Gathering formally arranged for a particular purpose, usually in a dedicated room and/or around a table.
memo,	Memo	Document	Document corresponding to a message or other information in writing sent by one person or department to another in the same Organization.
metal,	MetalTopic	BuildingMaterialTopic	25
metrology,	MetrologyBuildingTopic	ScienceAppliedToBuildingTopic	11
minutes,	Minutes	Document	Document containing of written record of what was said at a meeting.
mobile IP, mobile internet protocol,	MobileIPTopic	MobilePhoneTopic	
mobile phone architecture,	MobilePhoneArchitectureTopic	MobilePhoneTopic	
mobile phone frequency planning,	MobilePhoneFrequencyPlanningTopic	MobilePhoneTopic	
mobile phone planning system,	MobilePhonePlanningSystemTopic	MobilePhoneTopic	
mobile phone protocols,	MobilePhoneProtocolsTopic	MobilePhoneTopic	
mobile phones, cellular phones,	MobilePhoneTopic	TelecommunicationsTopic	Telecommunications part concerned with portable radiotelephones.
mobile phone satelite service,	MobilePhoneSateliteServiceTopic	MobilePhoneTopic	
mobile phone service,	MobilePhoneServiceTopic	MobilePhoneTopic	
mobile phone technology,	MobilePhoneTechnologyTopic	MobilePhoneTopic	
mobile phone terminals,	MobilePhoneTerminalsTopic	MobilePhoneTopic	
modification,	Modification	CorporateMemoryEvent	Corporate Memory Event corresponding to content transformed in the memory.
mortar,	MortarTopic	BuildingMaterialTopic	25
movie, film,	MovieRepresentation	AnimationRepresentation	Animation usually shown on a screen - cinema, TV, computer...- and often telling a story.
movie, talking movie,	TalkingMovieRepresentation	MovieRepresentation	Movie having spoken dialogue and/or soundtrack.
MP3, M.P.3, MP3 format,	MP3Format	MPEGFormat	MPEG Format of a audio documents encoded according to the third coding schemes of MPEG.
MPEG, M.P.E.G., M.P.E.G. format, moving picture experts group format,	MPEGFormat	DataFileFormat	Data File Format of animated images/movie compressed in Moving Picture Experts Group format.
multi-agents systems, MAS, M.A.S.,	MultiAgentSystemTopic	DistributedArtificialIntelligenceTopic	Distributed Artificial Intelligence where the systems are designed as organizations of autonomous and loosely coupled pieces of software.
music,	MusicTopic	AdditionalTopic	
musical language,	MusicalLanguage	Language	Natural Language representing musical notes.
narration, story, account,	Narration	Document	Document corresponding to an account describing incidents or events.Document la correspondant un compte decrivant des incidents ou des evenements.
national organization, national organisation group,	NationalOrganizationGroup	OrganizationGroup	Organization Group of nationwide scope, that is distribution throughout some Country of its members and/or activities.
natural language,	NaturalLanguage	Language	Language used by human beings, which has evolved naturally.
network,	NetworkTopic	TelecommunicationsTopic ComputerScienceTopic	A group of two or more computer systems linked together.
newcomer,	Newcomer	IntegrationProcessActor	Person newly arrived in the company.
news,	News	Document ExtractedDocument	Document about recent events.
newsgroup message, forum message,	NewsgroupMessage	Document	Document corresponding to messages displayed on the Internet and devoted to the discussion of a specified topic.
news letter,	NewsLetter	News ISSNHolderDocument	News issued to members of an Organization.
newspaper,	Newspaper	ISSNHolderDocument	Document regularly printed and consisting of news reports, articles, photographs and advertisements, and that is usually intended to be printed on a daily or weekly basis on large sheets of paper which are folded

			together but not permanently joined.
nomenclature,	Nomenclature	ReferenceDocument	Reference Document which gives a system for naming things, especially in a particular area of science.
non ferrous metal,	NonFerroMetalTopic	MetalTopic	25
non-linear filtering,	NonlinearFilteringTopic	FilteringTopic NonlinearSystemsTopic	
non-linear systems,	NonlinearSystemsTopic	MathematicsTopic	
non spatial entity,	NonSpatialEntity	Entity	Entity that has no spatial nature and does not pertain to space.
numberable entity,	NumberableEntity	RoleEntity	Entity to which one or more numbers are associated.
numerical analysis,	NumericalAnalysisTopic	AnalysisTopic	
object-programming,	ObjectProgrammingTopic	ProgrammingTopic	A type of programming in which programmers define not only the data type of a data structure, but also the types of operations -functions- that can be applied to the data structure.
observer,	Observer	TechnologyMonitoringActor	Technical monitoring actor required to monitor and filter all the technical and strategic documentation on innovative services they came across and send anything they judge as interesting input for the technology monitoring process to his area referent.
official document,	OfficialDocument	Document	Document agreed to or arranged by people in positions of authority.
ontology engineering,	OntologyEngineeringTopic	KnowledgeModelingTopic	Field dealing with the engineering of explicit, partial specification of a conceptualization.
order form,	OrderForm	Form	Form which a customer uses to request goods or a service.
organization, organisation,	Organization	OrganizationGroup	Organization Group including both informal and legally constituted organizations.
organizational entity, organisational entity,	OrganizationalEntity	ManagementAbleEntity ManageableEntity AdministrationAbleEntity ActivityAbleEntity InterestAbleEntity SituableEntity GroupAbleEntity GatheringEntity	Entity recognized by and within the organization.
organization chart,	OrganizationChart	Diagram	Diagram that graphically or in outline fashion depicts information about the control structure or resource use structure of an organization.
organization group, organisation group,	OrganizationGroup	OrganizationalEntity	An arganizational entity is composed of other Organization Entity working together in a structured way for a shared purpose.
organization homepage,	OrganizationalHomePage	Homepage	Homepage of an Organization s Web Site.
organization part,	OrganizationPart	OrganizationGroup	Organization Group which is a sub-organization of another Organization Group.
organization policy, corporate policy,	OrganizationPolicy	Manual	Manual that contains the terms of some policy of a particular organization.
outer layer, casing, enveloppe,	OuterLayerTopic	ShellTopic	222
outside development,	OutsideDevelopmentTopic	MaterialAndWorkTechnicsTopic	21
paper,	PaperMaterialTopic	BuildingMaterialTopic	25
paper,	Paper	DocumentaryMedium	Documentary Medium which is a thin flat material, made from crushed wood or cloth and used for writing, printing or drawing on.
parallel- object-programming,	ParallelObjectProgrammingTopic	ParallelProgrammingTopic ObjectProgrammingTopic	
parallel programming,	ParallelProgrammingTopic	ProgrammingTopic	A type of programming where processes occur simultaneously.
partition,	PartitionTopic	SecondWorkTopic	234
partner,	Partner	Organization	Organization with which an Organization is associated and collaborates. The cooperation can be limited (e.g. for the realization of a contract for a customer) or durable (mutual recognition of the evaluation procedures and test results, institutional partnership).
part time,	PartTime	ContractTypeAttribute	Contract type of an employee working part-time for an organization.
party event,	PartyEvent	SocialGathering EntertainmentEvent	Social Gathering at one location for people to communicate share some experience and to enjoy themselves.
patent,	Patent	OfficialDocument NumberableEntity	Official document that confers upon the creator of an invention the sole right to make, use, and sell that invention for a given period of time.
pathology,	PathologyTopic	MaterialAndWorkTechnicsTopic	26
pattern,	Pattern	NonSpatialEntity	Pattern of annotation or query.
payroll,	Payroll	Finance	Finance activity concerned with maintaining the list of the people employed, showing the total amount of money paid to the people employed by a particular company.
PDA, P.D.A., personal digital assistant,	PDATopic	MobilePhoneTerminalsTopic	
PDF, P.D.F.,	PDFFormat	DataFileFormat	Data File Format of a visual document encoded in Adobe Portable

P.D.F. format, Portable Document Format,			Document Format.
perception mode,	DocumentaryPerceptionMode	DocumentAttribute	Document Attribute giving the perception channel/way.
person, human, human being,	Person	ManageableEntity AdministrationAbleEntity ActivityAbleEntity LivingEntity InterestAbleEntity SituableEntity GroupAbleEntity GatheringEntity NumberableEntity	Living Entity belonging to mankind, an individual human being.
personal homepage,	PersonalHomePage	Homepage	Homepage of a Person s Web Site
Ph.D. student,	PhDStudent	Researcher Student	Student carrying supervised research usually based on at least 3 years graduate study and a dissertation to obtain a Ph.D./doctorate (the highest degree awarded by a graduate school).
physical entity,	PhysicalEntity	SpatialEntity	Spatial Entity made of matter.
physics,	PhysicsTopic	AdditionalTopic	Science of matter and energy and their interactions.
pipe and shaft, conduit, outer covering,	PipeAndShaftTopic	SecondWorkTopic	235
plaster,	PlasterTopic	CoatingTopic	25
pole,	Pole	OrganizationPart GroupOfIndividuals	Functional subdivision of a service. It has generally 2 to 8 people.
polymer,	PolymerTopic	BuildingMaterialTopic	25
postcard,	Postcard	PostMail	Post Mail consisting in a small, rectangular card, usually with a picture on one side and a message on the other, that can be sent without an envelope.
Post-doctorate,	PostDoctorate	Researcher	Student who recently finished his Ph.D. and has a contract with a limited duration to continue his search.
post mail, mail,	PostMail	Mail	Mail transmitted via the post office.
presentation,	Presentation	Document	Document presenting news or other information and intended to be broadcasted or printed.
private gathering,	PrivateGathering	Gathering	Gathering restricted to one particular group, not to other people.
probabilities, probabilistics,	ProbabilitiesTopic	MathematicsTopic	
proceedings,	Proceedings	Document	Document containing a written account of what transpired at a event and the documents presented by the actors of the event.
professional,	Professional	Person	Person who does activities that are characteristic of some job/profession/occupation for a livelihood.
profile,	Profile	Document	Document containing a biographical sketch.
programming,	ProgrammingTopic	ComputerScienceTopic	Science of organizing instructions so that, when executed, they cause the computer to behave in a predetermined manner.
programming language, programing language,	ProgrammingLanguage	ArtificialLanguage	Artificial Language for instructing a computer to perform specific tasks.
project group, project,	ProjectGroup	GroupOfIndividuals	Temporary group of individual working on a planned activity for a client with an associated fixed budget.
PS, P.S., P.S. format, postscript, Post Script Format,	PSFormat	DataFileFormat	Data File Format of a visual document encoded in Postscript format.
public document,	PublicDocument	DiffusionRight	The diffusion is not subjected to any restriction.
public gathering,	PublicGathering	Gathering	Gathering allowing anyone to see or hear what is happening.
pushed document trace,	PushedDocumentTrace	NonSpatialEntity	Such an element is created when the CoMMA push mode process retreive a document that can interests user.
quality assessment,	QualityAssessment	Service	Evaluation and certification of products and process.
rating value,	RatingValue	Attribute	Type of the feedback given after consulting a resource.
RDF, R.D.F., resource description framework,	RDFTopic	XMLTopic	Web Resource Description Framework.
record tape,	RecordTape	DocumentaryMedium	Documentary Medium where the recording is done on a magnetic tape.
reference document,	ReferenceDocument	Document	Document to which you can refer for authoritative facts.
report,	Report	Document	Document, usually a concise one, on a well defined topic and taking into account the identity of the readers.
representation system,	DocumentaryRepresentationSystem	DocumentAttribute	Document Attribute describing the system - signs, symbols, indices... - that serves as a means of expressing a Document Element in order to exhibit it to the mind.
research,	Research	Service	Service providing detailed study of a subject, esp. in order to discover -new- information or reach a -new- understanding.
research direction,	ResearchDirection	OrganizationPart	Organization Part responsible for a specific technical field e.g. mobile field.
researcher,	Researcher	Professional	Professional working in the Activity Field of Research, carrying out detailed

			study of a subject, esp. in order to discover - new - information or reach a - new - understanding.
research report,	ResearchReport	Report	Report presenting studies.
restoration, food,	Food	Service	Service of preparing and/or serving food.
RMI, R.M.I., remote method invocation,	RMITopic	NetworkTopic ObjectProgrammingTopic	Remote Method Invocation, a set of protocols being developed by Sun s JavaSoft division that enables Java objects to communicate remotely with other Java objects.
road,	RoadTopic	OutsideDevelopmentTopic	21
robotics,	RoboticsTopic	ArtificialIntelligenceTopic	branch of Artificial Intelligence concerned with the practical use of robots.
role entity,	RoleEntity	Entity	Entity that can play a role in a relation.
roof,	RoofTopic	ShellTopic	223
rubber,	RubberTopic	BuildingMaterialTopic	25
sale, sell,	Sell	Commerce	Commerce activity where a person or an organization gives a product or a service to another in return for money.
sales assistant, sales man, shop assistant, sales clerk,	Salesperson	Professional	Professional employed to sell merchandise to customers in a store or to customers that are visited.
sanitary equipment,	SanitaryEquipmentTopic	EquipmentTopic	242
scenario analysis,	ScenarioAnalysis	Document	Document based on available information about technologies, proposing potential medium term strategic scenarios. Reasonably there will be only a few reports of this kind in a year.
sciences applied to building,	ScienceAppliedToBuildingTopic	BuildingTopic	10
scientist,	Scientist	Professional	Professional educated and employed in one - or more - of the natural or abstract sciences.
sculpture, carving,	SculptureRepresentation	IconicRepresentationSystem	Iconic Representation System producing three-dimensional and usually tangible representations.
second work,	SecondWorkTopic	MaterialAndWorkTechnicsTopic	23
secretary,	Secretary	Assistant	Assistant who handles correspondence and clerical work for a boss or an organization.
security,	SecurityBuildingTopic	BuildingTransversalTopic	65
self employed,	SelfEmployed	ContractTypeAttribute	Contract type of an professional who earns a living from funds paid directly to him/her by customers, or who is paid by a company s/he owns. An Self-employed has no boss but him/herself.
service,	Service	Activity	Activity where a work is done by one person or group that benefits another.
service, service group,	ServiceGroup	OrganizationPart	Basic functional unit. A service is part of a department and is made up of several divisions or poles.
sexual attribute,	SexualAttribute	Attribute	Attribute representing either of the two categories - male or female - into which most organisms are divided.
shell,	ShellTopic	MaterialAndWorkTechnicsTopic	22
shutter,	ShutterTopic	JoineryTopic	231
sign language,	SignLanguage	NaturalLanguage	Natural Language expressed by visible hand gestures.
silent movie,	SilentMovieRepresentation	MovieRepresentation	Movie having no spoken dialogue and usually no soundtrack.
simulation,	SimulationTopic	ComputerScienceTopic MathematicsTopic	Science of theoretical account based on a similarity between the model and the phenomena that are to be explained.
single site organization,	SingleSiteOrganizationGroup	OrganizationGroup	Organization Group which has a single location as its physical quarters.
situable entity,	SituableEntity	RoleEntity SpatialEntity	Entity that can have a known location.
social gathering,	SocialGathering	Gathering	Formal Gathering of people who have the same or similar purposes in attending, and in which there is communication between the participants with sociability and maybe communal activities.
social ritual,	SocialRitual	SocialGathering	Social Gathering in which some kind of ritual is performed. E.g., a wedding, an awards ceremony, a baptism, an inauguration, a graduation ceremony, etc.
social science,	SocialScienceTopic	AdditionalTopic AdditionalTopic	30
software engineering,	SoftwareEngineeringTopic	ComputerScienceTopic	The computer science discipline concerned with developing computer applications.
soil mechanics,	SoilMechanicsBuildingTopic	ScienceAppliedToBuildingTopic	17
sound insulation, sound proofing,	SoundInsulationTopic	InsulationTopic	232
Spanish,	Spanish	NaturalLanguage	Natural Language spoken by the natives of Spain.
spatial entity,	SpatialEntity	Entity	Entity pertaining to or having the nature of space.
speech,	Speech	Document	Document corresponding to a formal talk given usually to a large number of people on a special occasion.
sports event,	SportsEvent	EntertainmentEvent	Entertainment Event based on sport activities.
spreadsheet, spread sheet,	Spreadsheet	Document	Document with multiple columns and rows to organize data for calculating and making adjustments based on new data.
stability,	StabilityBuildingTopic	ScienceAppliedToBuildingTopic	14
static image,	StaticImage	ImageRepresentation	Image not changing for a long time.
statistics,	StatisticsTopic	MathematicsTopic	Branch of applied mathematics concerned with the collection and interpretation of quantitative data and the use of probability theory to estimate population parameters.

stochastic dynamic systems,	StochasticDynamicSystemsTopic	DynamicSystemsTopic StatisticsTopic	
stone,	StoneMaterialTopic	BuildingMaterialTopic	25
storage format,	StorageFormat	DocumentAttribute	A particular arrangement to hold and retain data.
structure,	StructureTopic	ShellTopic	221
student,	Student	Person	Person who studies at an academic institution. This collection includes students at all levels of study in all types of educational institutions.
subsidiary,	Subsidiary	Organization	Organization that is completely controlled by another.
supervision authority,	SupervisionAuthority	Organization	Organization having the legal authority to supervise the activities of other organizations (ex. housing ministry).
symbolic learning, machine learning,	SymbolicLearningTopic	ArtificialIntelligenceTopic	Domain interested in methods enabling the computers to learn.
symbolic representation system,	SymbolicRepresentationSystem	DocumentaryRepresentationSystem	Representation System based on symbols that are associated with the objects they represent by a rule - e.g. the word "bird" and a real bird.
tactile,	TactilePerceptionMode	DocumentaryPerceptionMode	Documentary Perception Mode perceptible by touch.
TCP-IP, TCP/IP,	TCPIPTopic	InternetTopic	Transmission Control Protocol/Internet Protocol, the suite of communications protocols used to connect hosts on the Internet.
teach, teaching,	Teach	Education	Activity in which some people impart learned knowledge to others.
technical report,	TechnicalReport	Report	Report presenting technical results.
technician,	Technician	Professional	Professional trained in some specific technical processes.
technology monitoring actor,	TechnologyMonitoringActor	Person	A Person playing a role in technology monitoring process.
telecommunications,	TelecommunicationsTopic	AdditionalTopic	Domain concerned with the technology of electronic communication at a distance.
temporary,	Temporary	ContractTypeAttribute	Contract type of an employee working temporary for an organization.
test,	TestBuildingTopic	ScienceAppliedToBuildingTopic	11
test laboratory,	TestLaboratory	OrganizationPart	Group in charge of tests and attached to a service or a division.
text format,	TextFormat	DataFileFormat	Data file format storing data in ASCII.
thermal insulation,	ThermalInsulationTopic	InsulationTopic	232
thermics,	ThermicsBuildingTopic	ScienceAppliedToBuildingTopic	15
thesis,	Thesis	Document	Document corresponding to a long piece of writing on a particular subject advancing a new point of view resulting from research ; it is usually a requirement for an advanced academic degree.
thing, something, anything,	Something		Whatever exists animate, inanimate or abstraction.
time entity,	TimeEntity	Entity	Entity related to the continuum of experience in which events pass from the future through the present to the past.
time point,	TimePoint	TimeEntity	Time Entity corresponding to a particular, instantaneous point in time.
topology,	TopologyTopic	MathematicsTopic	Branch of pure mathematics that deals only with the properties of a figure X that hold for every figure into which X can be transformed with a one-to-one correspondence.
trainee,	Trainee	Student	Student who is being trained during an internship the organization.
training period,	TrainingPeriod	ContractTypeAttribute	Contract type of an employee who is learning and practicing the skills of a particular job.
training period report,	TrainingPeriodReport	Report	Report written by a trainee at the end of its training period. Rarely confidential, these reports deal with very restricted area.
transparency, slide,	Transparency	Document	Document of one page usually concise and prepared to be presented to a public by projection.
transparency show,	TransparencyShow	Document	Document corresponding to an ordered set of transparencies usually grouped around a topic.
transport,	Transport	Service	Service of providing transportation of goods or persons.
travel,	Travel	Service	Service giving information about prices and schedules for a trip and arranging tickets and accommodation.
trend analysis,	TrendAnalysis	Document	Synthesis Document written by an expert on the trends of a technological area.
tutor,	Tutor	IntegrationProcessActor	A person who gives private advice and instruction to a newcomer.
UML, U.M.L., unified modeling language,	UML	ArtificialLanguage	General-purpose notational Artificial Language for specifying and visualizing complex software, especially large, object-oriented projects.
UMTS, U.M.T.S., universal mobile telecommunications system,	UMTSTopic	MobilePhoneProtocolsTopic	
union,	Union	GroupOfIndividuals	Group of individuals formed to bargain with the employer.
unit,	Unit	GroupOfIndividuals	Group of individuals corresponding to a group of researchers focusing on a sub interest field e.g.: microwaves.
university,	University	Organization	Organization which does university-level teaching and/or research.
update, mise a jour,	Update	Modification	Modification corresponding to content changed into more recent one.
ventilation shaft,	VentilationShaftTopic	PipeAndShaftTopic	235
virtual reality,	VirtualReality	IconicRepresentationSystem	Iconic Representation System for simulating systems or environments with

			and within which people can interact.
virtual reality,	VirtualRealitySimulationTopic	ComputerGraphicsTopic SimulationTopic	A computer simulation of a real or imaginary system that enables a user to perform operations on the simulated system and shows the effects in real time.
visual,	VisualPerceptionMode	DocumentaryPerceptionMode	Documentary Perception Mode by sight.
visual perception,	VisualPerceptionTopic	BiologyTopic	
wap, WAP, W.A.P, wireless application protocol,	WapTopic	MobilePhoneProtocolsTopic	Wireless Application Protocol is a secure specification that allows users to access information instantly via handheld wireless devices such as mobile phones, pagers, two-way radios, smartphones and communicators.
water analyze and treatment,	WaterAnalyzeAndTreatmentBuildingTopic	ScienceAppliedToBuildingTopic	16
wav, WAV Format,	WAVFormat	DataFileFormat	Data File Format of a audio document encoded in the WAV format.
wave propagation,	WavePropagationTopic	PhysicsTopic	
Web,	WebTopic	NetworkTopic	Internet servers networks that support specially formatted documents.
web page, web site,	WebPage	Document	Document corresponding to a page on the World Wide Web.
web site,	WebSite	Document	Document made up of interconnected Web Pages, usually including a Homepage, generally located on the same server, and prepared and maintained as a collection of information by a person, group, or organization.
window,	WindowTopic	JoineryTopic	231
wml, W.M.L., wireless markup language,	WMLTopic	NetworkTopic MobilePhoneTopic	Wireless Markup Language is an XML language used to specify content and user interface for WAP devices.
Word, Word format, Microsoft Word format,	MSWordFormat	DataFileFormat	Data File Format of a visual document encoded in the format of Microsoft Word.
worker,	Worker	Professional	Professional member of the working class with a specific job.
XML, X.M.L., X.M.L. Format, Extensible Markup Language Format, format du Langage de Marqueurs Extensible,	XMLFormat	TextFormat	Text Format of a Documentary Element written in eXtensible Markup Language.
XML, X.M.L., extensible markup language,	XMLTopic	WebTopic	Web Extensible Markup Language enabling the definition, transmission, validation, and interpretation of data between applications and between organizations.
zip, zip format,	ZIPFormat	DataFileFormat	Data File Format for files compressed in ZIP format.

15.2 Lexicon view of relations

Terms	Property ID	Parent IDs	Domain	Range	Natural Language definition
address,	Address	SomeRelation	Location	Text	Address of a location.
administer,	Administer	SomeRelation	AdministrationAbleEntity	OrganizationalEntity	Relation denoting that an Entity -Domain- regulates the operations of an Organizational Entity -Range-.
alternative document,	AlternativeDocument	SomeRelation	Document	Document	Relation denoting that a document has other occurrences eg. different format, different URL...
assist,	Assist	SomeRelation	Tutor	Newcomer	Denotes that Tutor assists a Newcomer.
beginning,	Beginning	SomeRelation	Gathering	Text	Starting date/hour of an gathering.
birth date,	BirthDate	SomeRelation	Person	Text	Date of birth.
colleague,	Colleague	SomeRelation	Person	Person	one of a group of people who work together.
comments,	Comments	SomeRelation	Document	Text	Textual remark or observation about a document.
concern,	Concern	SomeRelation	EntityConcerningATopic	AdditionalTopic	Relation denoting that an entity (e.g.: a document, a gathering...) concerns a topic.
contain,	Contain	SomeRelation	Document	Document	Relation denoting that a document includes another one.
created by,	CreatedBy	SomeRelation	Document	ActivityAbleEntity	Relation denoting that a Document has been created by an Entity.
creation date,	CreationDate	SomeRelation	Document	Text	Date the document was created.
customer type,	CustomerType	SomeRelation	IndexCard	Text	type of clients to whom the service is addressed.
description of a pattern,	PatternDescription	SomeRelation	Pattern	Text	Textual remarks or explanations about a pattern.
designation,	Designation	SomeRelation	Something	Text	Identifying word or words by which a thing is called and classified or distinguished from others.
domain, thesis research domain,	HasForThesisDomain	SomeRelation	Thesis	Text	Domain associated given by the national nomenclature.
employed by,	EmployedBy	SomeRelation	Employee	OrganizationalEntity	Relation denoting that an Organization has an Employee working or doing a job for it and pays this Employee for it.
employment contract,	EmploymentContract	SomeRelation	Employee	ContractTypeAttribute	Relation denoting the Type of the contract that link an Employee to an organization.
end,	End	SomeRelation	Gathering	Text	Ending date of an gathering
extracted from,	ExtractedFrom	SomeRelation	Article	ExtractedDocument	Relation designating the document from which the article was extracted.
fax number,	FaxNumber	HasNumber	Location	Text	Fax number of a location.
first name, given name,	FirstName	Designation	Person	Text	The name that occurs first in a person s full name.
first visit,	FirstVisit	SomeRelation	ConsultationTrace	Text	Date of the first visit to a document.
geographical area,	GeographicalArea	SomeRelation	IndexCard	Text	geographical area in which the specific service or technology are developed or exist.
had for participant,	HadForParticipant	SomeRelation	Gathering	GatheringEntity	Relation denoting that an Organizational Entity participates/participated to a Gathering.
has for activity,	HasForActivity	SomeRelation	ActivityAbleEntity	Activity	Relation denoting that an Entity is carrying out an activity.
has for diffudion right,	HasForDiffusionRight	SomeRelation	Document	DiffusionRight	Relation denoting the diffusion right of a document.
has for favorite annotation,	HasForFavoriteAnnotation	SomeRelation	Person	Pattern	Relation denoting a favorite pattern of a user for annotating resources.
has for favorite query,	HasForFavoriteQuery	SomeRelation	Person	Pattern	Relation denoting a favorite pattern of a user for querying the base.
has for internal report number,	HasForInternalReportNumber	HasNumber	Report	Text	Internal report number.
has for medium,	HasForMedium	SomeRelation	Document	DocumentaryMedium	Relation denoting that a document uses a medium.
has for number,	HasNumber	SomeRelation	NumberableEntity	Text	Number associated to a 'numberable' entity.
has for ontological entrance point,	HasForOntologicalEntrancePoint	SomeRelation	Person	Something	Relation denoting a prefered entrance point for browsing the ontology.
has for origin,	HasForOrigin	SomeRelation	Document	DocumentOrign	Relation denoting the origin of a document.
has for perception mode,	HasForPerceptionMode	SomeRelation	Document	DocumentaryPerceptionMode	Relation denoting that a document uses a perception mode -audio, visual, tactile-.
has for personnal interest,	HasForPersonnalInterest	IsInterestedBy			Relation denoting that an entity has a personal interest
has for representation system,	HasForRepresentationSystem	SomeRelation	Document	DocumentaryRepresentationSystem	Relation denoting that a document uses a representation system.
has for storage format,	HasForStorageFormat	SomeRelation	Document	StorageFormat	Relation denoting that a document is stored in a given format.

has for work interest,	HasForWorkInterest	IsInterestedBy			Relation denoting that an Entity has a special work interest.
hiring date,	HireDate	SomeRelation	Employee	Text	The date when the employee was hired.
include,	Include	SomeRelation	OrganizationalEntity	GroupAbleEntity	Relation denoting that an Entity has as a part another Entity.
indication,	Indication	SomeRelation	Location	Text	Textual signs/clues pointing to the location e.g.: "in the cupboard of the rest room".
ISBN, International Standard Book Number,	HasForISBN	HasNumber	Book	Text	The International Standard Book Number (ISBN) is a system of numerical identification for books, pamphlets, educational kits, microforms, CD-ROM and braille publications.
is interested by,	IsInterestedBy	SomeRelation	InterestAbleEntity	AdditionalTopic	Relation denoting that an Entity is interested in a topic.
ISRN,	HasForISRN	HasNumber	Report	Text	Identification code for reports.
ISSN,	HasForISSN	HasNumber	ISSNHolderDocument	Text	Standardized international code which allows the identification of any serial publication.
issued on the occasion of,	IssuedOnTheOccasionOf	SomeRelation	Document	Gathering	Relation denoting that a document has been/is issued on the occasion of a gathering.
keyword,	Keyword	SomeRelation	Document	Text	Keyword representative of the content of the document.
last visit,	LastVisit	SomeRelation	ConsultationTrace	Text	Date of the last visit to a document.
manage, oversee, supervise, superintend,	Manage	SomeRelation	ManagementAbleEntity	ManageableEntity	Relation denoting that an entity is in charge/controls of another entity.
mobile number,	MobileNumber	HasNumber	Person	Text	Mobile phone number.
name, family name, surname, last name,	FamilyName	Designation	Person	Text	The name used to identify the members of a family.
network type,	NetworkType	SomeRelation	IndexCard	Text	ex: mobile, fixed, satellite
number, period number,	HasForPeriodNumber	HasNumber	ISSNHolderDocument	Text	(Period) number of a serial document.
organized by,	OrganizedBy	SomeRelation	Gathering	GatheringEntity	Relation denoting that an entity organizes / organized a gathering.
phone number,	PhoneNumber	HasNumber	Location	Text	Phone number of a location.
pushed document,	PushedDocument	SomeRelation	PushedDocumentTrace	Document	Document pushed by the CoMMA Push Mode.
rating given,	RatingGiven	SomeRelation	ConsultationTrace	RatingValue	User s feedback after a consultation. Can be like Good, Bad, etc. cf RatingValue.
RDF string,	RDFString	SomeRelation	Pattern	Text	String representing the RDF pattern.
refers monitoring to,	RefersMonitoringTo	SomeRelation	Observer	AreaReferent	Links an observer to the referent of his area.
related to,	RelatedTo	SomeRelation	AdditionalTopic	AdditionalTopic	Relation denoting that an Interest Field is linked to another.
reliability,	Reliability	SomeRelation	InformationForm	RatingValue	Reliability of the information.
see also,	SeeAlsoInformationSource	SomeRelation	InformationForm	Text	Other source of the information.
service provider,	ServiceProvider	SomeRelation	IndexCard	Text	indicates the provider of telecommunication services or technologies or terminals.
service type,	ServiceType	SomeRelation	IndexCard	Text	
situated,	Situated	SomeRelation	SituableEntity	Location	Relation denoting that an Entity is located in a Location.
sleeping partner,	HasForSleepingPartner	SomeRelation	Report	ActivityAbleEntity	Relation designating a sleeping partner of the report.
some relation,	SomeRelation		Something		An abstraction belonging to, linking, or characterising of two things.
source, information form source,	InformationFormSource	SomeRelation	InformationForm	Text	Source of the information.
summary,	Summary	SomeRelation	Document	Text	A account of the main points of a document.
supervised by,	HasForSupervisingOrganization	SomeRelation	Thesis	Organization	Organization (school, university, laboratory...) supervising the thesis.
target,	Target	SomeRelation	Document	InterestAbleEntity	Relation denoting that a Document is intended for some Entity.
technology,	Technology	SomeRelation	IndexCard	Text	
title,	Title	SomeRelation Designation	Document	Text	Designation of a document.
trainnee originating from,	HasForOriginatingOrganization	SomeRelation	TrainingPeriodReport	Organization	Organization (school, university, laboratory...) from where the trainee was originating.
type, thesis research type,	HasForThesisType	SomeRelation	Thesis	Text	Type associated given by the national nomenclature.
visit count,	VisitCount	SomeRelation	ConsultationTrace	Text	Number of visits to a document in the whole history of a profile.
visited document,	VisitedDocument	SomeRelation	ConsultationTrace	Document	Document concerned by this element of the use history of a profile.
visitor,	Visitor	SomeRelation	ConsultationTrace	Employee	Person concerned by this element of the history of a profile.

16 Tables of illustrations

16.1 List of figures

16.2 List of tables

17 References

[Abecker *et al.*, 1998] Abecker A., Bernardi A., Hinkelmann K., Kuhn O., and Sintek M., Toward a technology for organizational memories. IEEE Intelligent Systems, May/June 1998. http://citeseer.nj.nec.com/abecker98toward.html

[Ackerman and Halverson, 1998] Ackerman M.S. and C. Halverson (1998) "Considering an organization's memory," In Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work. pp. 39-48. Seattle, WA: ACM Press. http://citeseer.nj.nec.com/ackerman98considering.html

[Aguirre *et al.*, 2000] Aguirre, J.L., Brena, R. Cantu-Ortiz, F. (2000). Multiagent-based Knowledge Networks. To appear in the special issue on Knowledge Management of the journal Expert Systems with Applications.

[Alfines, 2000] "The process of Enterprise Development: Modeling, redesign, implementation and use" by Erlend Alfines; Chapter 11 pages 159-182 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[AMICE, 1993] AMICE, 1993 CIM-OSA Open System Architecture for CIM. 2d rev. ed. Berlin: Springer-Verlag.

[Andersen, 2000] "Enterprise Modeling for Business Process Improvement" by Bjørn Andersen; Chapter 10 pages 137-157 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Arens *et al.*, 1996] Y. Arens, C.A. Knoblock, W. Shen. Query reformulation for dynamic information integration. Journal of Intelligent Information

Systems, 6(2):99-130, 1996.

[Arisha *et al.*, 1999] K. Arisha, F. Ozcan, R. Ross, V. S. Subrahmanian, T. Eiter, and S. Kraus. IMPACT: A Platform for Collaborating Agents. IEEE Intelligent Systems, 14:64--72, March/April 1999.

[Armani *et al.*, 2000] Armani B., Bertino E., Catania B., Laradi D., Marin B., Zarri G.P., Repository Management in an Intelligent Indexing Approach for Multimedia Digital Libraries. In Proc. of the 12th Int. Symp. on Methodologies for Intelligent Systems (ISMIS), Charlotte, North Carolina, 2000.

[Arpírez *et al.*, 2001] WebODE: a Scalable Workbench for Ontological Engineering. Arpírez, J.C.; Corcho, O.; Fernández-López, M.; Gómez-Pérez, A. KCAP01. Victoria. Canada. October, 2001

[Ashish and Knoblock, 1997] Naveen Ashish and Craig A. Knoblock. Semi-automatic wrapper generation for internet information sources. In Proccedings of the Second IFCIS International Conference on Cooperative Information Systems (CoopIS), Charleston, SC, 1997.

[AUML] Agent Unified Modelling Language - AUML Home Page.

Available January 2001 at http://www.auml.org

[Aussenac, 1989] "Conception d'une méthodologie et d'un outil d'acquisition des connaissances expertes", Aussenac N., Thèse de Doctorat en informatique, Université P. Sabatier, Toulouse, Octobre 1989

[Aussenac-Gilles *et al.*, 2000] Aussenac-Gilles N., Biebow B, Szulman S (2000). Revisiting Ontology Design: a Method Based on Corpus Analysis, In Proc. EKAW'2000, Juan-les-Pins, p172-188.

[Austin, 1975] J. L. Austin, Marina Sbisa (Editor), J. O. Urmsson (Editor), How to Do Things with Words, Harvard Univ Pr, 2nd edition (December 1975); ISBN: 0674411528

[Bachimont, 2000] Bruno Bachimont: "Engagement sémantique et engagement ontologique: conception et réalisation d'ontologies en ingénierie des connaissances"; In "Ingénierie des connaissances Evolutions récentes et nouveaux défis", Jean Charlet, Manuel Zacklad, Gilles Kassel, Didier Bourigault; Eyrolles 2000, ISBN 2-212-09110-9

[Bachimont, 2001] Bruno Bachimont "Modélisation linguistique et modélisation logique des ontologies: l'apport de l'ontologie formelle." In Proceedings of IC 2001, pp 349-368 Plate-forme AFIA, Grenoble 25-28 juin 2001

[Barbuceanu and Fox, 1994] M. Barbuceanu and M.S.Fox, The Information Agent: An Infrastructure for Collaboration in the Integrated Enterprise, in Proc. of Cooperative Knowledge Based Systems, University of Keele, Staffordshire, UK, M. Deen (ed), DAKE Centre, Univ. of Keele, 1994

[Barthès, 1996] Barthès J.-P. ISMICK and Knowledge Management. In J. F. Schreinemakers ed, Knowledge Management: Organization, Competence and Methodology, Proc. of ISMICK'96, Rotterdam, the Neth., Wurzburg:Ergon Verlag, 21-22 Octobre 1996 p. 9-13.

[Bauer *et al.*, 2000] Bauer B., Müller J. P., Odell J., Agent UML: A formalism for specifying multiagent software systems. In P. Ciancarini and M. Wooldridge, editors, Agent-Oriented Software Engineering, Proceedings of the First International Workshop (AOSE-2000). Springer-Verlag: Berlin, Germany, 2000.

[Beauchène *et al.*, 1996] Beauchène, D., Mahé, S. et Rieu, C. – Enterprise Know-How: Capitalization and Benchmarking with an Enterprise Organizational Model. In J. F. Schreinemakers ed, Knowledge Management:

Organization, Competence and Methodology, Proc. of ISMICK'96, Rotterdam, the Netherlands, Wurzburg:Ergon, October 21-22, 1996, p. 194-206.

[Bellifemine *et al.*, 2001] F. Bellifemine, A. Poggi, G. Rimassa, Developing multi-agent systems with a FIPA-compliant agent framework. Software Practice & Experience, (2001) 31:103-128

See also JADE: Java Agent Development Framework.

Homepage available at http://sharon.cselt.it/projects/jade.

[Bergamaschi and Beneventano, 1999] Bergamaschi S., Beneventano D., Integration of Information from Multiple Sources of Textual Data, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet p53-77, Matthias Klusch, Springer 1999

[Bergenti and Poggi, 2000] Bergenti, B., Poggi, A. Exploiting UML in the Design of Multi-Agent Systems. In A. Omicidi, R. Tolksdorf, F. Zambonelli, eds., Engineering Societies in the Agents World - Lecture Notes on Artificial Intelligence, volume 1972, pp 106-113, 2000, Springer Verlag Publ, Berlin, Germany

[Berners-Lee *et al.*, 2001] Berners-Lee T., Hendler J., Lassila O., The Semantic Web, In Scientific American, May 2001, p35-43

[Berners-Lee, 1999] Berners-Lee T., W3C Issues Recommendation for Resource Description Framework (RDF), Introduces model for defining and organizing information, http://www.w3.org/Press/1999/RDF-REC

[Berney and Ferneley, 1999] Berney, B., Ferneley, E. (1999), "CASMIR: Information Retrieval Based on Collabo-rative User Profiling", In Proceedings of PAAM'99, pp. 41-56. www.casmir.net

[Bernus *et al.*, 96] Bernus, P.; Nemes, L.; and Williams, T. J. 1996. Architectures for Enterprise Integration. London: Chapman and Hall.

[Bertino *et al.*, 1999] Bertino E., Catania B., Zarri G.P., A Conceptual Annotation Approach to Indexing in a Web-Based Information System. In Proc. of the Int. Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, Santa Clara, California, 1999.

[Biezunski, 2001] Topic Maps at a glance, Michel Biezunski, http://www.infoloom.com/tmartic.htm

[Bond and Gasser, 1988] Bond A. H., Gasser L., Readings in Distributed Artificial Intelligence, San Mateo, CA: Morgan Kaufmann, 1988

[Bothorel and Thomas, 1999] Bothorel, C. and Thomas, H. (1999), "A Distributed Agent Based-Platform for Internet User Communities", In Proceedings of PAAM'99, Lancashire, pp. 23-40.

[Brachman *et al.*, 1989] Brachman R., Borgida A., McGuiness D. and Resnick L. The CLASSIC knowledge representation system. In proc. of the 11th International Joint Conference on Artificial Intelligence (IJCAI'89), Morgan-Kaufman, 1989.

[Brazier *et al.*, 1997] Brazier, F.M.T., Dunin Keplicz, B., Jennings, N., and Treur, J., DESIRE: Modelling Multi-Agent Systems in a Compositional Formal Framework. International Journal of Cooperative Information Systems, vol. 6, Special Issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems, (M. Huhns and M. Singh, eds.), 1997, pp. 67-94.

[Breuker and Van de Velde, 1994] Joost Breuker and Walter Van de Velde, editors. CommonKADS Library for Expertise Modeling: reusable problem solving components. IOS-Press/Ohmsha, Amsterdam/Tokyo, 1994.

[Brickley and Guha, 2000] Brickley D. et Guha R.V. – Resource Description Framework (RDF) Schema Specification. W3C Candidate Recommendation, 27 March 2000, http://www.w3.org/TR/rdf-schema/*

[Brickley and Guha, 2002] Brickley D., Guha R.V., RDF Vocabulary Description Language 1.0: RDF Schema, W3C Working Draft 30 April 2002, http://www.w3.org/TR/2002/WD-rdf-schema-20020430/

[Brooks, 1991] Brooks R.A., Intelligence without Representation. Artificial Intelligence, Vol.47, 1991, pp.139-159.

[Brusilovsky, 2001] P. Brusilovsky, Adaptive Hypermedia, User Modeling and User-Adapted Interaction 11: 87-110, 2001. Kluwer Academic Publishers.

[Burmeister, 1996] Burmeister B., Models and methodology for agent-oriented analysis and design. In K Fischer, editor, Working Notes of the KI'96 Workshop on Agent-Oriented Programming and Distributed Systems, 1996. DFKI Document D-96-06.

[Caire *et al.*, 2001] Caire G., Leal F., Chainho P., Evans R., Garijo F., Gomez J., Pavon J., Kearney P., Stark J., Massonet P., Agent oriented analysis using message/uml. In Agent-Oriented Software Engineering (AOSE), pages 101-108, Montreal, 2001.

[Carberry, 2001] S. Carberry, Techniques for Plan Recognition, User Modeling and User-Adapted Interaction 11: 31-48, 2001. Kluwer Academic Publishers.

[Caroll, 1997] "Scenario-Based Design", John M. Caroll, Chapter 17 "Handbook of Human-Computer Interaction" Second, completely revised edition, M. Helander, T.K. Landauer, P. Prabhu (eds), 1997, Elsevier Science B.V.

[Charlet *et al.*, 2000] Cours de DEA/DESS sur l'ingénierie des connaissances développé par Jean Charlet et Nathalie Aussenac-Gilles avec la participation de Bruno Bachimont et de Philippe Laublet.

[Chaudhri *et al.*, 1997] Chaudhri V., Farquhar A., Fikes R., Karp P. et Rice J., – The Generic Frame Protocol 2.0. 1997.

[Chein and Mugnier, 1997] Chein M., Mugnier M.L., Positive Nested Conceptual Graphs. In Proc. of the 5th International Conference on Conceptual Structures (ICCS'97), Seattle, WA, USA, LNAI 1257, Springer Verlag, p. 95-109, 1997.

[Chevallet, 1992] Chevallet J.P. In modele logique de Recherche d'Information applique au formalisme des graphes conceptuels. Le prototype ELEN et son experimentation sur un corpus de composants logiciels. Ph.D. thesis, Universite J. Fourier, Grenoble, 1992.

[Chevallet, 1994] Chevallet J.P. Utilisation des graphes conceptuels pour des systèmes de recherche d'informations orientes vers la précision des réponses. In actes des journees "Graphes Conceptuels" du PRC-GDR IA, LIRMM, Montpellier, p. 35-53, 1994.

[Coleman *et al.*, 1994] Coleman D., Arnold P., Bodo S., Dollin C., Gilchrist H., Hayes F., and Jeremaes P., Object-Oriented Development: The Fusion Method., Prentice Hall, 1994.

[Collet *et al.*, 91] C. Collet, M. N. Huhns, and W. Shen. Resource integration using a large knowledge base in CARNOT. IEEE Computer, pages 55-62, December 1991

[Collinot *et al.*, 1996] Collinot A., Drogoul A., Benhamou P., Agent oriented design of a soccer robot team. In Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS-96), pages 41–47,

Kyoto, Japan, December 1996.

[CoMMA, 2000] CoMMA Consortium: Philippe PEREZ, Hervé KARP Atos Integration; Rose DIENG, Olivier CORBY, Alain GIBOIN, Fabien GANDON INRIA; Joel QUINQUETON LIRMM; Agostino POGGI, Giovanni RIMASSA University of Parma; Claudio FIETTA CSELT; Juergen MUELLER, Joachim HACKSTEIN T-Nova, Corporate Memory Management through Agents, In Proceedings E-Work & E-Business, Madrid, Octobre 17-20, 2000, pp 383-406

[Conrad *et al.*, 1997] Stefan Conrad Can Türker Gunter Saake . Towards Agent-Oriented Specification of Information Systems. Proceedings of the first International Conference on Autonomous Agents 97, Marina Del Rey, California USA 0 1997 ACM 0-99791-977-0/97/02

[Corby *et al.*, 2000] Corby O., Dieng R., Hébert C., A Conceptual Graph Model for W3C Resource Description Framework. In Proc. ICCS'2000 Darmstadt Germany

[Corby and Faron-Zucker, 2002] Corby O., Faron-Zucker C., Corese: A Corporate Semantic Web Engine, Workshop on Real World RDF and Semantic Web Applications 11th International World Wide Web Conference 2002 Hawaii

[Corcho and Gómez-Pérez, 2000] Corcho O. and Gómez-Pérez A. – A Roadmap to Ontology Specification Languages. In R. Dieng et O. Corby (éd.), Knowlege Engineering an Knowledge Management Methods, Models, and Tools. Poceedings of EKAW 2000, 2000, pp.80-96.

[Cyc, 2000] Online august 2000 version of the Upper Cyc® Ontology
http://www.cyc.com/cyc-2-1/cover.html
Cyc® is a Registered Trademark of Cycorp, Inc., 3721 Executive Center Drive, Suite 100 Austin, TX 78731

[Davenport, 1996] T.H. Davenport, Some Principles of Knowledge Management, Posted April 10,1996. Knowledge Management Server Graduate School of Business, University of Texas at Austin,
http://www.bus.utexas.edu/kman

[Davis and Smith, 1983] Davis, R., and Smith, R. G. 1983. Negotiation as a Metaphor for Distributed Problem Solving. Artificial Intelligence 20(1): 63–100.

[Decker and Sycara, 1997] K. Decker, K.P. Sycara., Intelligent adaptive information agents. Journal of Intelligent Information Systems, 9(3):239-260, 1997.

[Decker *et al.*, 1998] Decker S, Brickley D, Saarela, Angele. A Query Service for RDF. Query Languages 98, W3C Workshop.

[Decker *et al.*, 1999] Decker S., Erdmann M., Fensel D., Studer R., Ontobroker: Ontology based access to distributed and semi-structured information. In Meersman et al Semantic Issues in Multimedia, Systems, Kluwer

[Delteil *et al.*, 2001] Delteil, Faron, Dieng, Extension of RDF(S) based on the CGs Formalisms, in H.S. Delugach, G. Stumme (ed), Conceptual Structures: Broadening the Base. Proceedings of 9th International Conference on Conceptual Structures, ICCS 2001, Stanford, CA, USA, July 30-August 3, 2001, Springer-Verlag, LNAI 2120, p. 275 - 389.

[Delteil *et al.*, 2001b] Delteil A., Faron C., Dieng R., Learning Ontologies from RDF Annotations, in Proceedings of the IJCAI Workshop "Ontology Learning", Seattle, 2001.

[Demazeau, 1995] Demazeau Y., From Interactions to Collective Behaviour in Agent-Based Systems, I European Conference on Cognitive Science,

Saint-Malo, p. 117-132. 1995

[Deng, 2000] "Modeling for Performance Measurement" by Wei Deng Solvang; Chapter 4 pages 87-106 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Dieng *et al.*, 1998] "Building of a Corporate Memory for Traffic Accident Analysis.", R. Dieng, A. Giboin, C. Amergé, O. Corby, S. Després, L. Alpay, S. Labidi, S. Lapalut., In AI Magazine, Vol. 19, n.4, p. 80-100, Winter 1998.

[Dieng *et al.*, 1999] R. Dieng, O. Corby, A. Giboin, M. Ribière. Methods and Tools for Corporate Knowledge Management. S. Decker and F. Maurer eds, International Journal of Human-Computer Studies, special issue on knowledge Management, vol. 51, pp. 567-598, 1999. Academic Press

[Dieng *et al.*, 2001] Methodes Et Outils Pour La Gestion Des Connaissances: Une approche pluridisciplinaire du Knowledge Management (2nd Edition), Rose Dieng-Kuntz - Olivier Corby - Fabien Gandon - Alain Giboin - Joanna Golebiowska - Nada Matta - Myriam Ribière, Dunod Edition - INFORMATIQUES Série Systèmes d'information - ISBN 2 10 006300 6

[Dieng, 1990] "Méthodes et outils d'acquisition des connaissances." by Rose Dieng, Technical report from INRIA n° 1319, Novembre 1990.

[Dieng, 1993] "Méthodes et outils d'acquisition des connaissances." Rose Dieng; In J.-Cl. Spérandio, editor, "L'ergonomie dans la conception des projets informatiques" pages 335-411. OCTARES, Toulouse, August 1993.

[Dieng, 1995] Dieng R., Specifying a cooperative system through agent-based knowledge acquisition. In Proceedings of the International Workshop on Cooperative Systems (COOP'95), pages 141–160, Juan-les-Pins, January 1995.

[Doyle and Hayes-Roth, 1998] Doyle P., Hayes-Roth B., Agents in Annotated Worlds, In Proceedings of the Second Annual Conference on Autonomous Agents p173-180, Minneapolis, MN USA1998, ACM Press / ACM SIGART

[Drogoul and Ferber, 1992] [Drogoul and Ferber, 1992] Drogoul A., Ferber J., "From Tom-Thumb to the Dockers: Some Experiments with Foraging Robots" in From Animals to Animats II, MIT Press, Cambridge, pp. 451-459, 1992.

[Drucker, 1994] Post Capitalist Society by Peter F. Drucker, Ed. HarperBusiness; ISBN: 0887306616; Reprint edition (May 1994)

[Ducourneau *et al.*, 1998] Langages et modèles à Objets Etat des recherches et perspectives Ducourneau, Euzenat, Masini, Napoli INRIA, Collection Didactique, 1998, ISSN 0299 - 0733; ISBN 2 - 7261 - 1131

[Dunin-Keplicz and Treur, 1995] Dunin-Keplicz B., Treur J., Compositional formal specification of multi-agent systems. In M. Wooldridge and N. R. Jennings, editors, Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890), pages 102--117. Springer-Verlag: Berlin, Germany, January 1995.

[Durfee and Lesser, 1989] Durfee, E. H., and Lesser, V. 1989. Negotiating Task Decomposition and Allocation Using Partial Global Planning. In Distributed Artificial Intelligence, Volume 2, eds. L. Gasser and M. Huhns, 229–244. San Fran-cisco, Calif.: Morgan Kaufmann.

[Dzbor *et al.*, 2000] M. Dzbor, J. Paralic and M. Paralic, Knowledge Management in a Distributed Organisation, In Proc. of the BASYS'2000 - 4th IEEE/IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing, Kluwer Academic Publishers, London, September 2000, ISBN 0-7923-7958-6, pp. 339-348

[Elammari and Lalonde, 1999] Elammari M., Lalonde W., An Agent-Oriented Methodology: High-Level and Intermediate Models, proceedings of Agent-Oriented Information Systems 1999, Heidelberg, Germany, 14-15 June 1999.

[Ermine, 1998] Ermine J. L., Capter et créer le capital savoir, Annales des Mines. p. 82-86, Novembre 1998.

[Ermine, 2000] J.L. Ermine, Challenges and Approaches for Knowledge Management in Companies, Workshop Knowledge Management: Theory and Applications, Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases, September 13-16, 2000, Lyon, France

[Etzioni and Weld, 1994] A Softbot-Based Interface to the Internet, Etzioni, O., Weld, D., Communication of the ACM, Special Number on Intelligent Agents, July 1994, pp.72-76.

[Etzioni and Weld, 1995] Etzioni O., Weld D.S., Intelligent Agents on the Internet: Fact, Fiction and Forecast, University of Washington, IEEE Expert/Intelligent Systems & Their Applications Vol. 10, No. 4, August 1995

[Euzenat, 1996] Euzenat, J. (1996) . Corporate memory through cooperative creation of knowledge bases and hyper-documents. In B. Gaines, M. Musen eds, Proc of KAW'96, Banff, Canada, November, pp. 36-1 36-18.

[Fagerhaug, 2000] "Enterprise Modeling for Self-Assessment" by Tom Fagerhaug; Chapter 8 pages 107-118 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Fan, 2000] Fan. X., Towards a Building Methodology for Software Agents. TUCS Technical Report, Turku Centre for Computer Science, No 351, June 2000. http://citeseer.nj.nec.com/fan00towards.html

[Farquhar *et al.*, 1996] Farquhar A., Fikes R. et Rice J. – The Ontolingua Server: a Tool for Collaborative Ontology Construction. In B. Gaines, M. Musen eds, Proc of KAW'96, Banff, Canada, novembre 1996, pp. 44-1 - 44-19. Dans http://ksi.cpsc.ucalgary.ca/KAW/KAW96/KAW96Proc.html

[Fensel *et al.*, 1998] Fensel D., Decker S., Erdmann M. and Studer R. Ontobroker: Or How to Enable Intel-ligent Access to the WWW. In B. Gaines, M. Musen eds, Proc of the 11th Workshop on Knowledge Acquisition, Modeling and Management (KAW'98), Banff, Canada, April 18-23, 1998. http://ksi.cpsc.ucalgary.ca/KAW/KAW98/KAW98Proc.html

[Fensel *et al.*, 1999] Fensel D., Angele J., Decker S., Erdmann M., Schnurr H., Staab S., Studer R., Witt A. (1999). On2broker: Semantic-Based Access to Information Sources at the WWW. In Proceedings of the World Conference on the WWW and Internet (WebNet 99), Honolulu, Hawai, USA, October 25-30, 1999.

[Fensel *et al.*, 2000] Fensel D., Horrocks I., Van Harmelen F., Decker S., Erdmann M. et Klein M. – OIL in a Nutshell. In R. Dieng et O. Corby (éd.), Knowlege Engineering an Knowledge Management Methods,

Models, and Tools. Poceedings of EKAW 2000, 2000, pp.1-16.

[Feray *et al.*, 1998] Feray, N., Ngomo M., Villefranche, L., Trupin, E., Pecuchet, J.-P., Lecourtier Y. Mne-mosNet: a corporate memory system for the research laboratories, Proc. of IC'98, 13-15 May 1998, Pont-à-Mousson, France, pp. 251-256.

[Ferber and Gutknecht, 1998] Ferber J., Gutknecht O., A meta-model for the analysis and design of organizations in multi-agent systems. IEEE Computer Society, Proc. 3rd ICMAS, p. 128-135, 1998.

[Ferber, 1999] J. Ferber, Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence, Addison-Wesley, 1999, ISBN: 0201360489

[Ferguson, 1995] Ferguson, I.A. Integrated control and coordinated behaviour: a case for agent models. In M. Wooldrige and N.R. Jennings, editors. Intelligent Agents: Theories, Architectures and Languages. LNAI-890, pp. 203-218, Springer Verlag, Berlin 1995.

[Fernandez *et al.*, 1997] M. Fernandez, A. Gomez-Perez, and N. Juristo. METHONTOLOGY: From Ontological Arts Towards Ontological Engineering. In Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering, Stanford, USA, pages 33-40, March 1997.

[Fillion *et al.*, 1995] Fillion, F.; Menzel, C.; Blinn, T.; and Mayer, R. 1995., An Ontology-Based Environment for Enterprise Model Integration. Paper presented at the IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing, 19–20 August, Montreal, Quebec, Canada.

[Finin *et al.*, 1994] Finin, T.; Fritzson, R.; McKay, D.; and McEntire, R. 1994. KQML as an Agent Com-munication, Language. In Proceedings of the Third International Conference on Information and Knowledge Management, (CIKM94), 456–463. New York: Association of Computing Machinery.

[Finin *et al.*, 1998] Finin T., Nicholas C. and Mayfield J., Agent-based information retrieval tutorial, ADL'98. http://www.csee.umbc.edu/abir/

[FIPA] Foundation for Intelligent Physical Agents, FIPA Specifications

http://www.fipa.org/

[Fischer, 2001] G. Fischer, User Modeling in Human-Computer Interaction, User Modeling and User-Adapted Interaction 11: 65-86, 2001, Kluwer Academic Publishers

[Foner, 1997] Leonard N. Foner . Yenta: A Multi-Agent, Referral-Based Matchmaking System. Proceedings of the first International Conference on Autonomous Agents 97, Marina Del Rey, California USA 0 1997 ACM 0-99791-977-0/97/02

[Fox and Gruninger, 1998] Fox, M.S., Gruninger, M., (1998), "Enterprise Modelling", AI Magazine, AAAI Press, Fall 1998, pp. 109-121.

[Fox *et al.*, 1993] Fox, M.; Chionglo, J. F.; and Fadel, F. G. 1993. A Commonsense Model of the Enterprise. In Proceedings of the Second Industrial Engineering Research Conference, 425-429. Norcross Ga.: Institute for Industrial Engineers.

[Franklin and Graesser, 1996] Stan Franklin, Art Graesser . Is it an Agent, or just a Program ? : A Taxonomy for Autonomous Agents. Institute for Intelligent Systems University of Memphis, Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages, Springer - Verlag, 1996

[Fraser, 1994] Fraser, J. Managing Change through Enterprise Models, in Proc. of

Expert Systems'94, the 14th Annual Conference of the British Computer Society Specialist Group on Expert Sys-tems, Cambridge UK, December.

[Fukuda, 1995] Y. Fukuda Variations of Knowledge in Information Society, In Proceedings ISMICK 95, p3-8

[Gandon, 2002a] F. Gandon, Ontology Engineering: a survey and a return on experience, Research Report of INRIA n°4396, France, March 2002

[Gasser *et al.*,1987] Gasser L., Braganza C., Herman N., MACE: a flexible testbed for distributed AI research, In M.N. Huhns, editor, Distributed Artificial, Intelligence, pages 119-152, 1987.

[Gasser, 1991] Gasser L., Social Conceptions of Knowledge and Action: DAI Foundations and Open Systems, Artificial Intelligence 47, 107-138, 1991

[Gasser and Huhns, 1989] Gasser L., Huhns M., Distributed Artificial Intelligence 2, San Mateo, CA: Morgan Kaufmann, 1989

[Gastinger and Szegheo, 2000] "Enterprise Modeling Approaches" Almuth Gastinger, Orsolya Szegheo; Chapter 5 pages 55-69 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Genesereth and Fikes, 1992] Genesereth M. and Fikes R. – Knowledge Interchange Format. Technical Report Computer Science Department. Standford University, Logic 92-1, 1992.

[Genesereth and Ketchpel, 1994] Genesereth, M. R., and Ketchpel, S. P., "Software Agents," Communication of the ACM, Vol. 37, No. 7 July 1994.

[Genesereth *et al.*, 1997] M. Genesereth, A. Keller, O. Duschka, Infomaster: An Information Integration System, in proceedings of 1997 ACM SIGMOD Conference, May 1997.

[Gerbé, 2000] Gerbé, Un modèle uniforme pour la modélisation et la métamodélisation d'une mémoire d'entreprise, PhD Thesis Université de Montréal, Janvier 2000

[Giboin *et al.*, 2002] Giboin A., Gandon F., Corby O., Dieng R., Assessment of Ontology-based Tools: Systemizing the Scenario Approach, EKAW 2002, Workshop, EON - Evaluation of Ontology-based Tools, Madrid, September 2002

[Glaser, 1996] Glaser N., Contribution to Knowledge Modelling in a Multi-Agent Framework (the Co-MoMAS Approach). PhD thesis, L'Universtité Henri Poincaré, Nancy I, France, November 1996.

[Glaser, 2002] Glaser N., Multiagent Systems, Artificial Societies, And Simulated Organizations, Volume 4, Kluwer Academic Publishers, 2002, ISBN: 1402070616

[Golebiowska, 2002] Exploitation des ontologies pour la mémoire d'un projet-véhicule: Méthode et outil SAMOVAR, Joanna Golebiowska, Ph. D. Computer Science Thesis, University of Nice Sophia Antipolis, 4th of February 2002.

[Gómez-Pérez *et al.*, 1996] Gómez-Pérez, A.; Fernandez, M.; De Vicente, A. Towards a Method to Conceptualize Domain Ontologies Workshop on Ontological Engineering. ECAI'96. 1996. Pages 41-51

[Greengrass, 2000] E. Greengrass, Information Retrieval: A Survey, 30/11/2000, http://www.csee.umbc.edu/cadip/readings/IR.report.120600.book.pdf

References

[Gruber, 1993] Gruber, T.R. (1993). Toward Principles for the Design of Ontologies Used for Knowledge Sharing, In Formal Ontology in Conceptual Analysis and Knowledge Representation, edited by Nicola Guarino and Roberto Poli, Kluwer Academic Publishers

[Grundstein and Barthès, 1996] Grunstein, M. and Barthès J.-P. (1996). An Industrial View of the Process of Capitalizing Knowledge. In J. F. Schreinemakers ed, Knowledge Management: Organization, Competence and Methodology, Proc. of ISMICK'96, Rotterdam, the Netherlands, Wurzburg:Ergon Verlag, Advances in Knowledge Management, vol. 1, October 21-22, p. 258-264.

[Grundstein, 1995] Grundstein M. (1995). La capitalisation des connaissances de l'entreprise, système de production de connaissances. L'entreprise apprenante et les Sciences de la Complexité. Aix-en-Provence, Mai.

[Guarino and Giaretta, 1995] Guarino N., Giaretta P., Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N. J. I. Mars (ed.), Towards Very Large Knowledge Bases, IOS Press 1995.

[Guarino and Welty, 2000] Guarino, N. and Welty, C. 2000. Towards a methodology for ontology-based model engineering. In Proceedings of ECOOP-2000 Workshop on Model Engineering. Cannes, France. Available from http://www.ladseb.pd.cnr.it/infor/ontology/Papers/OntologyPapers.html

[Guarino and Welty, 2002] Guarino, N. and Welty, C. 2002. Evaluating Ontological Decisions with OntoClean. Communications of the ACM, 45(2): 61-65.

[Guarino *et al.*, 1999] Guarino N., Masolo C., and Vetere G. Ontoseek: Content-based access to the web. In IEEE Intelligent Systems, vol. 14 (3), p. 70-80, 1999.

[Guarino, 1992] Guarino N. Concepts, Attributes, and Arbitrary Relations: Some Linguistic and Ontological Criteria for Structuring Knowledge Bases. Data and Knowledge Engineering 8: 249-261, 1992.

[Guarino, 1997] Guarino N.,Understanding, Building and Using Ontologies. A Commentary to "Using Explicit Ontologies in KBS Development", by van Heijst, Schreiber, and Wielinga. International Journal of Human and Computer Studies vol. 46 n. 2/3, pp. 293-310, 1997

[Guttman *et al.*, 1999] Guttman R., Moukas A., Maes P., Agent as Mediators in Electronic Commerce, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet, p131-152, Matthias Klusch, Springer 1999

[Haton *et al.*, 1991] Haton J.P., Bouzid N, Charpillet F., Haton M.C., Lâasri B., Lâasri H., Marquis P., Mondot T., Napoli A., Le Raisonnement en Intelligence Artificielle, InterEditions, ISBN 2729603352, 1991

[Hayden *et al.*, 1999] Hayden S.C., Carrick C., Yang Q., A Catalog of Agent Coordination Patterns, In Proceedings of the Third Annual Conference on Autonomous Agents p412-413, Seattle, WA USA MAY 1-5, 1999 Edited By Oreb Etzioni; Jörg P. Müller; Jeffrey M. Bradshaw ACM Press / ACM SIGART

[Hayes, 2002] Hayes P., RDF Model Theory, W3C Working Draft 29 April 2002

http://www.w3.org/TR/2002/WD-rdf-mt-20020429/

[Heflin *et al.*, 1998] Heflin J., Hendler J., Luke S. Reading between the lines: Using SHOE to discover implicit knowledge from the Web. In proc. of the AAAI Workshop on Artificial Intelligence and Information Integration. WS-98-14. AAAI Press, p. 51-57, 1998.

[Heflin *et al.*, 1999] Heflin J., Hendler J., Luke S., SHOE: A Knowledge Representation Language for Internet Applications. Institute for Advanced Computer Studies, University of Maryland at College Park.

[Hewitt, 1986] Hewitt, C. 1986. Offices Are Open Systems. ACM Transactions of Office Automation Systems, 4(3): 271–287.

[Hofstadter, 1999] Hofstadter D. R., Gödel, Escher, Bach: An Eternal Golden Braid, Basic Books, ISBN: 0465026567, 20th anniv edition, January 1999.

[Huhns, 1998] Huhns M.N., Agent Foundations for Cooperative Information Systems. In: Proc. s of the Third International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technology; London 1998; Edited by H.S. Nwana and D.T. Ndumu.

[Iglesias *et al.*, 1998] Iglesias C. A., Garijo M., González J. C., Velasco J. R., Analysis and design of multiagent systems using MAS-CommonKADS. In AAAI'97 Workshop on Agent Theories, Architectures and Languages, Providence, RI, July 1997. ATAL. (An extended version of this paper has been published in INTELLIGENT AGENTS IV: Agent Theories, Architectures, and Languages, Springer Verlag, 1998.

[Iglesias *et al.*, 1999] C. Iglesias, M. Garijo, and J. Gonzales. A survey of agent-oriented methodologies. In J. Müller, M. Singh, and A. Rao, editors, Intelligent Agents V. Proceedings of the Fifth International Workshop on Agent Theories, Architectures, and Languages (ATAL-98), Lecture Notes in Articial Intelligence Vol. 1555, pages 317-330. Springer-Verlag, 1999.

[ITU-T, 1994] ITU-T. Z100. CCITT specification and description language (SDL). Technical report, ITU-T, June 1994.

[Jacobson, 1992] Jacobson I., Object-Oriented Software Engineering: A Use Case Driven Approach, Addison-Wesley, ISBN: 0201544350, 1992

[Jasper *et al.*, 1999] Jaspers R., Uschold M., Kitzmiller T. - Integrated Framework for Knowledge Management. In Debenham, Decker, Dieng, Macintosh, Matta, Reimer eds, Electronic Proceedings of IJCAI'99 Workshop on Knowledge Management and Organizational Memories, Stockholm, Sweden, 31 juillet 1999, http://www.inria.fr/acacia/WORKSHOPS/IJCAI99-OM/proceedings.html

[Jennings *et al.*, 1996] Jennings N. R., Mamdani E. H., Laresgoiti I., Perez J., and Corera J., Using ARCHON to develop real-world DAI applications. IEEE Expert 11 (6) 64-70, 1996.

http://citeseer.nj.nec.com/jennings96using.html

[Jennings, 2000] Jennings N. R., "On Agent-Based Software Engineering", Artificial Intelligence, Vol. 117, No. 2, pp. 277-296, 2000, Elsevier Science

[Jonker *et al.*, 2000] Jonker, C.M., Klusch, M., and Treur, J., Design of Collaborative Information Agents. In: M. Klusch, and L. Kerschberg (eds.), Cooperative Information Agents IV, Proceedings of the Fourth International Workshop on Cooperative Information Agents, CIA 2000. Lecture Notes in Artificial Intelligence, vol. 1860, Springer Verlag, 2000, pp. 262-283. http://citeseer.nj.nec.com/jonker00design.html

[Karp *et al.*, 1999] Karp R., Chauddri V. et Thomere J. – XOL: An XML-Based Ontology Exchange Language, 1999

[Kassel *et al.*, 2000] The project is presented in "Construction et exploitation d'une ontologie pour la gestion des connaissances d'une équipe de

recherche" Kassel G., Abel M.-H., Barry C., Boulitreau P., Irastorza C., Perpette S.; Proceedings of IC'2000, Toulouse, 10-12 Mai 2000, available at http://www.irit.fr/IC2000/actes-enligne.html

The ontology is accessible online at

[Kassel, 2002] Kassel G., OntoSpec : une méthode de spécification semi-informelle d'ontologies. In Actes des Journées Francophones d'Ingénierie des Connaissances: IC'2002, Rouen, May 2002, p75-87

[Kayser, 1997] La représentation des connaissances, Daniel Kayser, ISBN 2866016475, 1997

[Kemerling] Philosophy Pages from Garth Kemerling: This site offers helpful information for students of the Western philosophical tradition.

Available January 2001 at http://www.philosophypages.com/

[Kendall *et al.*, 1996] Kendall E. A., Malkoun M. T., Jiang C., A methodology for developing agent based systems for enterprise integration. In D. Luckose and Zhang C., editors, Proceedings of the First Australian Workshop on DAI, Lecture Notes on Artificial Intelligence. Springer-Verlag: Heidelberg, Germany, 1996.

[Kendall, 1999] Kendall E., Role Modeling for Agent System Analysis, Design, and Implementation. In First International Symposium on Agent Systems and Applications ASA'99, Third International Symposium on Mobile Agents MA'99, Palm Springs, October, 1999.

[Kifer *et al.*, 1995] Kifer M., Lausen G. et Wu J. – Logical Foundations of Objetc-Oriented and Frame-Based Languages. Journal of the ACM, 1995.

[Kimble *et al.*, 2001] C. Kimble, P. Hildreth and P. Wright - Communities of Practice: Going Virtual, Chapter 13 in Knowledge Management and Business Model Innovation, Idea Group Publishing, Hershey (USA)/London (UK), 2001. pp 220 - 234. ISBM 1 878289 98 5

[Kinny *et al.*, 1996] Kinny D., Georgeff M., Rao A., A methodology and modelling technique for systems of BDI agents. In W. van der Velde and J. Perram, editors, Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World MAAMAW'96, (LNAI Volume 1038). Springer-Verlag: Heidelberg, Germany, 1996.

[Kirk *et al.*,1995] The Information Manifold, T. Kirk, A. Y. Levy, Y. Sagiv, and D. Srivastava. The Information Manifold. In Proc. of the AAAI Spring Symposium on Information Gathering in Distributed Heterogeneous Environments. 1995

[Kiss and Quinqueton, 2001] A. Kiss, J. Quinqueton, Multiagent Cooperative Learning of User Preferences, Proc. of European CMLP & PKDD, 2001

[Klusch, 2001] Klusch, M. Information Agent Technology for the Internet: A Survey, Journal on Data and Knowledge Engineering, Special Issue on Intelligent Information Integration, D. Fensel (Ed.), Vol. 36(3), Elsevier Science, 2001

[Klusch, 1999a] M. Klusch, Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet, Springer, pages IX-XVII, 1999

[Klusch, 1999b] Introduction of Part 1 of the book "Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet", Matthias Klusch, Springer, 1999

[Klusch, 1999c] Klusch M., Introduction of Part 2 of the book Intelligent Information Agent: Agent-Based Information Discovery and

Management on the Internet, p127-130, Springer, 1999

[Klusch, 1999d] Foreword of the book "Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet", Matthias Klusch, Springer, pages V-VII, 1999

[Knight and Luke, 1994] Knight K. and Luke S. Building a large Knowledge Base for Machine Translation. Proc. Amer. Assoc. Artificial Intelligence Conf. (AAAI-94). AAAI Press, pages 773-778, Melon Park, California. 1994.

[Kobsa *et al.*, 1999] Kobsa, A., Koenemann, J. and Pohl, W.: 1999, Personalized hypermedia presentation techniques for improving online customer relationships. Technical report No. 66 GMD, German National Research Center for Information Technology, St. Augustin, Germany.

[Kobsa, 2001] A Kobsa, Generic User Modeling Systems, User Modeling and User-Adapted Interaction 11: 49-63, 2001. Kluwer Academic Publishers. Printed in the Netherlands.

[Korfhage, 1997] R. Korfhage, Information Storage and Retrieval, John Wiley & Sons (ed), 1997, ISBN: 0471143383

[Kosanke, 1993] Kosanke K., CIMOSA - A European Development for Enterprise Integration. IOS Press, 1993.

[Kuhn and Abecker, 1997] Kuhn, O. and Abecker, A. (1997) "Corporate Memories for Knowledge Management in Industrial Practice: Prospects and Challenges", Journal of Universal Computer Science, Vol. 3, no.8, pp.929-954.

[La France, 1992] "Questioning Knowledge Acquisition" By Marianne La France; In "Questions And Information Systems" Edited by: Lauer, Thomas W, Peacock Eileen, Graesser, Arthur C.; Editions: L. Erlbaum associates; 1992;ISBN: 0-8058-11018-8.

[Lassila and Swick, 1999] Lassila O. and Swick R. – Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, 22 February 1999, http://www.w3.org/TR/REC-rdf-syntax/

[Le Bortef, 1994] Le Bortef G. - De la compétence: Essai sur un attracteur étrange. Les Éditions d'Organisation, 1994.

[Lesser and Corkill, 1983] Lesser V. R. and Corkill D. D., The distributed vehicle monitoring testbed: A tool for investigating distributed problem solving networks. In AI Magazine, 4(3), 15--33, 1983.

[Levenshtein, 1966] Levenshtein I. V., Binary Codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory, 10(8):707–710, 1966.

[Levy *et al.*, 1995] A.Y. Levy, D. Srivastava, T. Kirk, Data model and query evaluation in global information systems. Journal of Intelligent Information Systems 5(2):121-143, September 1995

[Liao *et al.*, 1999] Minghong Liao, Knut Hinkelmann, Andreas Abecker, and Michael Sintek. A Competence Knowledge Base System for the Organizational Memory. In: Frank Puppe (ed.) XPS-99 / 5. Deutsche Tagung Wissensbasierte Systeme, Würzburg, Springer Verlag, LNAI 1570, March 1999.

[Lieberman, 1999] Lieberman H., Personal Assitants for the Web: An MIT Perspective. In Klusch Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet. pp. 279-292 Springer

[Loke and Davison, 1998] Loke S. W. and Davison A. LogicWeb: Enhancing the Web with

Logic Programming. Journal of Logic Programming, 36:195-240, 1998

[Luck *et al.*, 1997] Luck M., Griffiths N., d'Inverno M., From agent theory to agent construction: A case study. In J. P. Müller, M. Wooldridge, and N. R. Jennings, editors, Intelligent Agents III (LNAI Volume 1193), pages 49-64. Springer-Verlag: Berlin, Germany, 1997.

[Luke and Heflin, 2000] Luke S., Heflin J. – SHOE 1.01 Proposed specification. SHOE Project. February, 2000, http://www.cs.umd.edu/projects/plus/SHOE/

[Macintosh, 1994] Macintosh A., Corporate Knowledge Management State-of-the-Art Review, (1994), In Proc. ISMICK 94, p. 131-146

[Maedche and Staab, 2001] Maedche A. and Staab S., Comparing Ontologies — Similarity Measures and a Comparison Study, Internal Report No. 408, Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany, March 2001

[Manola and Miller, 2002] Manola F., Miller E., RDF Primer, W3C Working Draft 26 April 2002, http://www.w3.org/TR/rdf-primer/

[Marchiori and Saarela, 1998] Marchiori M. and Saarela J. Query+Metadata+Logic=Metalog. In proc. of the W3C Query Language Workshop, 1998. http://www.w3.org/TandS/QL/QL98/pp.html

[Mark *et al.*, 2002] Mark G., Gonzalez V., Sarini M., Simone C., Reconciling different perspectives: an experiment on technology support for articulation, in (eds) Blay-Fornarino M., Pinna-Dery A. M., Schmidt K., Zaraté P., Proceedings of COOP 2002, Cooperative Systems Design, IOS Press, p23-37

[Martin, 1996] Martin P. Exploitation de Graphes Conceptuels et de documents structurés et Hypertextes pour l'acquisition de connaissances et la recherche d'informations. Thèse de Doctorat en Informatique, Université de Nice - Sophia Antipolis, 14 Octobre 1996

[Martin and Eklund, 1999] "Knowledge Indexation and Retrieval and the Word Wide Web" Martin Ph. & Eklund P. (2000b). IEEE Intelligent Systems, special issue "Knowledge Management and Knowledge Distribution over the Internet", May/June 2000.
http://meganesia.int.gu.edu.au/~phmartin/WebKB/doc/papers/ieee99/ieee99.ps
Ontology accessible online:
http://www-sop.inria.fr/acacia/personnel/phmartin/RDF/phOntology.html

[Martin and Eklund, 2000] Martin P. and Eklund P. Knowledge retrieval and the world wide web. In R. Dieng ed, IEEE Intelligent Systems, Special Issue on Knowledge Management and Knowledge Distribution Over the Internet, p. 18-25, 2000.

[Maurer and Dellen, 1998] Maurer F., and Dellen B. A Concept for an Internet-based Process-oriented Knowledge Management Environment. In B. Gaines, M. Musen eds, Proc of the 11th Workshop on Knowledge Acquisition, Modeling and Management (KAW'98), April 18-23.

[Mena *et al.*, 1996] E. Mena, V. Kashyap, A. Sheth and A. Illarramendi, "OBSERVER: An approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies," Proceedings of the 1st IFCIS International Conference on Cooperative Information Systems (CoopIS '96), Brussels, Belgium, June 1996

[Minsky, 1986] Minsky M. The Society of Mind. Simon and Schuster, New York, 1986.

[Mizoguchi and Ikeda, 1997] Riichiro Mizoguchi and Mitsuru Ikeda. Towards Ontology Engineering, In Proceedings of The Joint 1997 Pacific Asian Conference on Expert systems / Singapore International Conference on Intelligent Systems, pp. 259-266, 1997

[Mizoguchi *et al.*, 1997] Roles of Shared Ontology in AI-ED Research -- Intelligence, Conceptualization, Standardization, and Reusability -- Riichiro Mizoguchi, Mitsuru Ikeda, and Katherine Sinitsa Proc. of AIED-97, pp.537-544, also as Technical Report AI-TR-97-4, I.S.I.R., Osaka University, 1997

[Molina *et al.*, 1997] Molina, G., Papakonstantinou, Y., Quass, D., Rajarman, A., Sagiv, Y., Ullman, J., Widom, J. The SIMMIS approach to mediation: Data models and languages. Journal of Intelligent Information Systems, 8(2):117-132, 1997.

[Moulin and Brassard 1996] Moulin B, Brassard M., A scenario-based design method and an environment for the development of multiagent systems. In D. Lukose and C. Zhang, editors, First Australian Workshop on Distributed Artificial Intelligence, (LNAI volume 1087), pages 216–231. Springer-Verlag: Heidelberg, Germany, 1996.

[Müller, 1996] Müller H. J., Towards agent systems engineering. International Journal on Data and Knowledge Engineering. Special Issue on Distributed Expertise, (23):217–245, 1996.

[Muslea *et al.*, 1999] Muslea I., Minton S., Knoblock C., A Hierarchical Approach to Wrapper Induction, In Proceedings of the Third Annual Conference on Autonomous Agents, p190-197, Seattle, WA USA MAY 1-5, 1999 Edited By Oreb Etzioni; Jörg P. Müller; Jeffrey M. Bradshaw, ACM Press / ACM SIGART

[Nagendra Prasad and Plaza, 1996] Nagendra Prasad M.V. N., Plaza E. - Corporate Memories as Distributed Case Libraries. Dans B. Gaines, M. Musen eds, Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96), Banff, Alberta, Canada, 9-14 novembre 1996, p. 40-1 40-19. Also in http://ksi.cpsc.ucalgary.ca/KAW/KAW96/KAW96Proc.html

[Newell, 1982] A. Newell, The knowledge level, Artificial Intelligence 18 (1982) 87–127.

[Nielsen, 2001a] J. Nielsen, (2001). Ten Usability Heuristics for Interface Design, in “useit.com: Jakob Nielsen's Website”, Available at: http://www.useit.com/papers/heuristic/heuristic_list.html

[Nielsen, 2001b] Nielsen, J. (2001b). Severity Ratings for Usability Problems, in “useit.com: Jakob Nielsen's Website”, Available at: http://www.useit.com/papers/heuristic/severityrating.html

[Nodine *et al.*, 1999] M. Nodine, J. Fowler, T. Ksiezyk, B. Perry, M. Taylor, A. Unruh, Active Information Gathering In Infosleuth™; In Proc. Internat. Symposium on Cooperative Database Systems for Advanced Applications, 1999

[Nonaka and Konno, 1998] I. Nonaka and N. Konno, The concept of 'Ba': building a foundation for knowledge creation, California Management Review, V. 40, #3, 1998

[Nonaka and Takeuchi, 1995] Nonaka I., Takeuchi H. - Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation, Oxford University Press, 1995

[Nonaka, 1991] Nonaka I., The knowledge-creating company. Harvard Business Review. Novembre-Décembre 1991, pp. 96-104 ou p. 312-320.

[Nonaka, 1994] Nonoka I. Dynamic Theory of Organizational Knowledge Creation.

Organizational Science, Vol. 5, No. 1. Février 1994, p. 14-37.

[Nwana and Ndumu, 1999] Nwana H. S. and Ndumu D. T., A Perspective on Software Agents Research, In: The Knowledge Engineering Review, Vol 14, No 2, pp 1-18, 1999.

[Nwana, 1996] Hyacinth S. Nwana, Software agents: an overview, The Knowledge Engineering Review, Vol. 11:3, pages 205-244, 1996

[O'Leary, 1998] D. O'Leary, "Enterprise Knowledge Management," Computer, Vol. 31, No. 3, March 1998, pp. 54–61.

[Odell *et al.*, 2000] Odell J., Van Dyke Parunak H., Bauer B., Representing agent interaction protocols in UML. In P. Ciancarini and M. Wooldridge, editors, Agent-Oriented Software Engineering, Proceedings of the First International Workshop (AOSE-2000). Springer-Verlag: Berlin, Germany, 2000.

[Odubiyi *et al.*, 1997] J. Odubiyi, D. Kocur, S. Weinstein, N. Wakim, S. Srivastava, C. Gokey, J. Graham. Scalable Agent-based Information Retrieval Engine. In Proceedings of the First Annual Conference on Autonomous Agents. Marina del Rey, California USA February 5-8, 1997 ACM Press / ACM SIGART, p. 292-299.

[Ounis and Pasca, 1998] Ounis I. and Pasca M., RELIEF: Combining expressiveness and rapidity into a single system. Accepted at the ACM SIGIR'98 conference, Melbourne, Australia, 1998. http://citeseer.nj.nec.com/ounis98relief.html

[Özsu and Valduriez, 1999] Özsu M.T., Valduriez P., Principles of Distributed Database Systems, 2nd edn., Prentice-Hall, 1999.

[Panzarasa and Jennings, 2001] Panzarasa P., Jennings N.R., The organisation of sociality: A manifesto for a new science of multi-agent systems, Proc. MAAMAW'01, LNAI, Springer Verlag, 2001.

[Papazoglou and Heuvel, 1999] Papazoglou M.P., Heuvel W. From Business Process to Coopeartive Information Systems: An Information Agents Perspective, p. 10-36, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet", Matthias Klusch, Springer, 1999

[Poitou, 1995] Poitou J.P., Documentation is Knowledge: An Anthropological Approach to Corporate Knowledge Management. In J. P. Barthès ed, Proceedings of the Third International Symposium on the Management of Industrial and Corporate Knowledge (ISMICK'95), Compiègne, octobre 1995, pp. 91-103.

[Polanyi, 1966] Polanyi M. - The Tacit Dimension. London: Routledge & Kegan Paul, 1966.

[Pomian, 1996] J. Pomian, Mémoire d'entreprise techniques et outils de la gestion du savoir, Ed. Sapientia, 1996, ISBN 2911761006

[Purvis *et al.*, 2000] Purvis M, Cranefield S., Bush G., Carter D., McKinlay B., Nowostawski M. Ward R., The NZDIS Project: an Agent-Based Distributed Information Systems Architecture, Proceedings of the Hawai'i International Conference On System Sciences, January 4-7, 2000, Maui, Hawaii.

[Rabarijaona *et al*., 1999] A. Rabarijaona and R. Dieng and O. Corby, Building a XML-based Corporate Memory. In John Debenham and Stefan Decker and Rose Dieng and Ann Macintosh and Nada Matta and Ulrich Reimer eds, Proc. of the IJCAI'99 Workshop on Knowledge Management and Organizational Memories, Stockholm, Sweden, 31 juillet 1999.

[Rabarijaona *et al.*, 2000] "Building a XML-based Corporate Memory ", Rabarijaona A., Dieng R., Corby O., Ouaddari R., IEEE Intelligent Systems, Special

Issue on Knowledge Management and Internet, May-June 2000, p56-64

[Ralescu and Fadlalla, 1990] Ralescu A. L.and Fadlalla A., The Issue of Semantic Distance in Knowledge Representation with Conceptual Graphs, AWOCS 90, 141--142

[Rao and Georgeff, 1991] Rao A. S. and Georgeff M. P., Modeling Agents Within a BDI Architecture . In: R. Fikes and E. Sandewall (eds.), Proc. of the 2rd International Conference on Principles of Knowledge Representation and Reasoning (KR'91), pp. 473-484, Cambridge, Mass., April 1991. Morgan Kaufmann.

[Ribière, 1999] Ribiere M. Représentation et gestion de multiples points de vue dans le formalisme des graphes conceptuels. PhD from Univ. Nice Sophia-Antipolis

[Rolstadås, 2000] "Development trends to support Enterprise Modeling" by Asbjørn Rolstadås; Chapter 1 pages 3-16 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Røstad, 2000] "Enterprise Modeling for Enterprise Integration" by Carl Christian Røstad; Chapter 9 pages 119-136 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Rumbaugh *et al.*, 1991] Rumbaugh J., Blaha M., Premerlani W., Eddy F., Lorensen V., Object-Oriented Modeling and Design., Prentice-Hall, 1991.

[Rupolph *et al.*, 1996] Rupolph E., Grabowski J., Graubmann P., Tutorial on message sequence charts (MSC). In Proceedings of FORTE/PSTV'96 Conference, October 1996.

[Sabas *et al.*, 2002] Sabas A., Delisle S., Badri M., A Comparative Analysis of Multiagent System Development Methodologies: Towards a Unified Approach, Third International Symposium From Agent Theory to Agent Implementation, at the 16th European Meeting on Cybernetics and Systems Research, 2002, Vienna, Austria, p. 599-604

[Salvat and Mugnier, 1996] E. Salvat, M.L. Mugnier. Sound and complete forward and backward chainings of graph rules, In Proc. of the 4th ICCS'96, Sydney, Australia, LNCS 1115, Springer-Verlag, p. 248-262, 1996.

[Sangüesa Sol and Pujol Serra, 1999] NetExpert: A multiagent system for Expertise Location, Ramon Sangüesa Sol, Josep M. Pujol Serra, IJCAI'01 Workshop on Knowledge Management and Organizational Memories (KM/OM), p85-93

[Scapin and Bastien, 1997] D.L. Scapin and J.M.C. Bastien, (1997), Ergonomic criteria for evaluating the ergonomic quality of interactive systems, Behaviour & Information Technology, vol 16, no. 4/5, pp.220-231.

[Scheer, 1989] Scheer, A-W. 1989. Enterprise-Wide Data Modeling: Information Systems in Industry. New York: Springer-Verlag.

[Schlenoff *et al.*, 1996] Schlenoff, C.; Knutilla, A.; Ray, S. 1996. Unified Process Specification Language: Requirements for Modeling Process, Interagency Report 5910, National Institute of Standards and Technology, Gaithersburg, Maryland.

[Searle, 1969] Searle J., Speech Acts - An Essay in the Philosophy of Language, Cambridge University Press 1969.

[Sebillotte, 1991] "Décrire des tâches selon les objectifs des opérateurs de l'Interview

	à la Formalisation" by Suzanne Sebillotte; Technical report from INRIA n°125; 1991.
[Seely Brown and Duguid, 1998]	Organizing Knowledge, John Seely Brown and Paul Duguid. California Management Review. Spring 1998, Vol. 40, No. 3; pp. 90-111.
[Shoham, 1993]	Shoham Y., 1993. Agent-Oriented Programming. Artificial Intelligence 60(1): 51–92.
[Simon and Grandbastien, 1995]	Simon G. and Grandbastien M., Corporate knowledge: a case study in the detection of metallurgical flaws. In J. P. Barthès ed, Proceedings of the Third International Symposium on the Management of Industrial and Corporate Knowledge (ISMICK'95), Compiègne, France, pp. 43-52.
[Simon, 1996]	Simon, G. (1996). Knowledge Acquisition and Modeling for Corporate Memory: Lessons learnt from Experience. In B. Gaines, M. Musen eds, Proc. of KAW'96, Banff, Canada, November, pp. 41-1 41-18.
[Singh and Huhns, 1999]	Singh M.P., Huhns M. N., Social Abstraction for Information Agents, In the book Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet p37-52 Matthias Klusch Springer 1999
[Solberg, 2000]	"Enterprise Modeling and Education" by Claus Solberg; Chapter 12 pages 183-199 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1
[Sowa, 1984]	Conceptual Structures: Information Processing in Mind and Machine, J.F. Sowa, Addison-Wesley, 1984.
[Sowa, 2000a]	"Ontology, Metadata, and Semiotics", John F. Sowa, Proceedings of ICCS'2000 in Darmstadt, Germany, on August 14, 2000. Published in B. Ganter & G. W. Mineau, eds., Conceptual Structures: Logical, Linguistic, and Computational Issues, Lecture Notes in AI #1867, Springer-Verlag, Berlin, 2000, pp. 55-81.
[Sowa, 2000b]	Guided Tour of Ontology; John F. Sowa http://www.jfsowa.com/ontology/guided.htm
[Sowa, 2002]	Conceptual Graphs Standard, J.F. Sowa ISO/JTC1/SC 32/WG2 N 000 http://users.bestweb.net/~sowa/cg/cgstand.htm
[Staab and Maedche, 2000]	Staab, Steffen and Maedche, Alexander[2000]. Axioms are Objects, too -Ontology Engineering beyond the Modeling of Concepts and Relations. Internal Report 399, Institute AIFB, Karlsruhe University.
[Steels, 1993]	Steels L., Corporate Knowledge Management. In Barthès Proc. ISMICK'93 pp. 9-30
[Stephanidis, 2001]	C. Stephanidis, Adaptive Techniques for Universal Access, User Modeling and User-Adapted Interaction 11: 159-179, 2001, Kluwer Academic Publishers.
[Sveiby, 1997]	Tacit Knowledge, Karl E. Sveiby, Dec 31 1997, http://www.sveiby.com.au/articles/Polanyi.html
[Sycara *et al.*, 1999b]	K. Sycara, J. Lu, M. Klusch, S. Widoff, Matchmaking among Heterogeneous Agents on the Internet. In Proceedings of the 1999 AAAI Spring Symposium on Intelligent Agents in Cyberspace, Stanford University, USA 22-24 March 1999.
[Sycara, 1998]	The Many Faces of Agents, K. P. Sycara, AI Magazine, 19(2):

Summer 1998, 11-12

[Sycara, 1998b] Multiagent Systems, K. P. Sycara, AI Magazine, 19(2): Summer 1998, 79-92

[Sycara, 1999a] K. Sycara, "In-Context Information Management through Adaptive Collaboration of Intelligent Agents", p53-77 in the book "Intelligent Information Agent: Agent-Based Information Discovery and Management on the Internet", Matthias Klusch, Springer, 1999

[Szegheo and Martinsen, 2000] "Modeling and Simulation" by Orsolya Szegheo, Kjetil Martinsen; Chapter 14 pages 217-230 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Szegheo and Petersen, 2000] "Modeling of the Extended Enterprise" by Orsolya Szegheo, Sobah Abbas Petersen; Chapter 15 pages 232-246 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Szegheo, 2000] "Introduction to Enterprise Modeling" by Orsolya Szegheo; Chapter 3 pages 21-32 of "Enterprise Modeling: Improving Global Industrial Competitiveness"; Edited by Asbjørn Rolstadås and Bjørn Andersen; Kluwer Academic Publisher; 2000; ISBN 0-7923-7874-1

[Tourtier, 1995] Tourtier P.-A. - Analyse préliminaire des métiers et de leurs interactions. Rapport intermédiaire du projet GENIE, INRIA-Dassault-Aviation, 1995.

[TOVE, 2000] Online august 2000 version of the TOVE ontology
http://www.eil.utoronto.ca/tove/ontoTOC.html

From the Enterprise Integration Laboratory (EIL), Department of Industrial Engineering, University of Toronto, 4 Taddle Creek Rd., Toronto, Ontario M5S 3G9 http://www.eil.utoronto.ca/eil.html

[Troncy and Isaac, 2002] Troncy R., Isaac A. - DOE: une mise en oeuvre d'une méthode de structuration différentielle pour les ontologies. In 13ièmes Journées Francophones d'Ingénierie des Connaissances, IC'2002, 28-30 May 2002, Rouen, France.

[Uschold and Gruninger, 1996] M. Uschold and Gruninger M. Ontologies: Principles, methods and applications. Knowledge Engineering Review, Vol. 11:2, 93-136, 1996. Also available as AIAI-TR-191 from AIAI, The University of Edinburgh.

[Uschold *et al.*, 1998] " The Enterprise Ontology " Mike Uschold, Martin King, Stuart Moralee and Yannis Zorgios (1998), The Knowledge Engineering Review, Vol. 13, Special Issue on Putting Ontologies to Use (eds. Mike Uschold and Austin Tate). Also available from AIAI as AIAI-TR-195 http://www.aiai.ed.ac.uk/project/enterprise/

[Van Elst and Abecker, 2001] Van Elst, Abecker, Domain Ontology Agents in Distributed Organizational Memories In Proc. Workshop on Knowledge Management and Organizational Memories, IJCAI, 2001.

[Van Heijst *et al.*, 1996] Van Heijst G, Van der Spek R. et Kruizinga E. - Organizing Corporate Memories. Dans B. Gaines, M. Musen eds, Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'96), Banff, Canada, novembre 1996, pp. 42-1 42-17. Also in
http://ksi.cpsc.ucalgary.ca/KAW/KAW96/KAW96Proc.html

[Varga *et al.*, 1994] Varga L. Z., Jennings N. R., Cockburn D., Integrating intelligent systems into a cooperating community for electricity distribution management. International Journal of Expert Systems with Applications, 7(4):563–579, 1994.

[Verharen, 1997] Verharen E. M., A Language-Action Perspective on the Design of Cooperative Information Agents. PhD thesis, Katholieke Universieit Brabant, the Netherlands, March 1997.

[Vernadat, 1996] "Enterprise modeling and integration: principles and applications" Vernadat F.B.; Editions Chapman & Hall, London 1996

[von Krogh, 1998] Care in Knowledge Creation: Special Issue of Knowledge and the Firm by Georg von Krogh, California Management Review, Vol. 40, No. 3, p. 133, March 22, 1998, ISSN: 0008-1256.

[Wagner, 2000] Wagner G., Agent-Oriented Analysis and Design of Organizational Information Systems, Proceedings of Fourth IEEE International Baltic Workshop on Databases and Information Systems, May 2000, Vilnius (Lithuania).

[Webb *et al.*, 2001] G. I. Webb, M. J. Pazzani and D. Billsus, Machine Learning for User Modeling, User Modeling and User-Adapted Interaction 11: 19-29, 2001., 2001 Kluwer Academic Publishers.

[Weggeman, 1996] M. Weggeman. Knowledge Management: The Modus Operandi for a Learning Organization. In J. F. Schreinemakers ed, Knowledge Management: Organization, Competence and Methodology, Proc. of ISMICK'96, Rotterdam, the Netherlands, Wurzburg:Ergon Verlag, Advances in Knowledge Management, vol. 1, 21-22 Octobre 1996, p. 175-187.

[Weinstein *et al.*, 1999] P.C. Weinstein, W.P. Birmingham, E.H. Durfee. Agent-Based Digital Libraries: Decentralization and Coordination. IEEE Communication Magazine, pp. 110-115, 1999

[Welty and Ide, 1999] Welty C and Ide N. Using the right tools: enhancing retrieval from marked-up documents. In Journal of Computers and the Humanities, 33(10), p. 59-84, 1999.

[Welty and Jenkins, 2000] Welty C. and Jenkins J. Untangle: a new ontology for card catalog systems. In proc. of the National Conference on Artificial Intelligence (Demo Abstract), H. Kautz and B. Porter eds., (AAA'00). AAAI Press, 2000.

[Wharton *et al.*, 1994] Wharton, C., Rieman, J., Lewis, C., and Polson, P. (1994). The Cognitive Walkthrough Method: A Practitioner's Guide. In Usability Inspection Methods, J. Nielsen and R.L. Mack (Eds.), New York: John Wiley & Sons, pp.105-141.

[Wiederhold, 1992] Wiederhold, G. 1992. Mediators in the architecture of future information systems. IEEE Computer 25, 3 (March), 38-49.

[Williams, 1991] Williams, T. J., and the Members of the Industry–Purdue University Consortium for CIM. 1991. The PURDUE Enterprise Reference Architec-ture, Technical report, 154, Purdue Laboratory for Applied Industrial Control, Purdue University.

[Wirfs-Brock *et al.*, 1990] Wirfs-Brock R., Wilkerson B., Wiener L., Designing Object-Oriented Software., Prentice-Hall, 1990.

[Witten and Frank, 1999] Witten, I.H., Frank, E. Data Mining – Practical Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, San Francisco, Ca, 1999

[Wooldridge and Ciancarini, 2000] M. Wooldridge, P. Ciancarini, Agent-Oriented Software Engineering: The State of the Art, AOSE 2000, June 10, 2000,

Lecture Notes in Computer Science 1957 Springer 2001, ISBN 3-540-41594-7, 1-28

[Wooldridge and Jennings, 1995] Wooldridge M., Jennings N.R., Intelligent Agents: Theory and Practice. The Knowledge Engineering Review, 10(2): pp. 115-152

[Wooldridge *et al.*, 1999] Wooldridge M., Jennings N.R., Kinny D. A Methodology for Agent-Oriented Analysis and Design. In Proc of Autonomous Agents '99 Seattle, ACM 1-58113-066-x/99/05

[Wooldridge, 1997] M. Wooldridge, Agent-based software engineering, IEEE Proc. Software Engineering 144 (1) (1997) 26–37.

[Zacklad and Grundstein, 2001a] M. Zacklad, M Grundstein (Eds.), Management des connaissances Modèles d'entreprises et applications, Traité IC2. Paris, Hermès. 2001. ISBN 2-7462-0235-2

[Zacklad and Grundstein, 2001b] M. Zacklad, M Grundstein, Ingénierie et Capitalisation des connaissances, Traité IC2. Paris, Hermès. 2001. ISBN 2-7462-0234-4

[Zambonelli *et al.*, 2000] F. Zambonelli, N.R. Jennings, and M. Wooldridge. "Organisational Abstractions for the Analysis and Design of Multi-Agent Systems". In Proc. of the 1st International Workshop on Agent-Oriented Software Engineering at ICSE 2000, Limerick, Ireland, June 2000

[Zarri *et al.*, 1999] Zarri G.P., Bertino E., Black B., Brasher A., Catania B., Deavin D., Di Pace L., Esposito F., Leo P., J.McNaught, Persidis A., Rinaldi F., and Semeraro G., CONCERTO, an environment for the intelligent indexing, querying and retrieval of digital documents. In LNAI 1609: Foundations of Intelligent Systems, Proc. of the 11th Int. Symp. ISMIS'99, pages 226-234, June 1999. Warsaw, Poland.

[Zhong *et al.*, 2002] Zhong J., Zhu H., Li J., Yu Y., Conceptual Graph Matching for Semantic Search, ICCS 2002, p92-106

[Zukerman and Albrecht, 2001] I. Zukerman and D. W. Albrecht, Predictive Statistical Models for User Modeling, User Modeling and User-Adapted Interaction 11: 5-18, 2001, Kluwer Academic Publishers.

DISTRIBUTED ARTIFICIAL INTELLIGENCE AND KNOWLEDGE MANAGEMENT: ONTOLOGIES AND MULTI-AGENT SYSTEMS FOR A CORPORATE SEMANTIC WEB

ABSTRACT:

This work concerns multi-agents systems for the management of a corporate semantic web based on an ontology. It was carried out in the context of the European project CoMMA focusing on two application scenarios: support technology monitoring activities and assist the integration of a new employee to the organisation. Three aspects were essentially developed in this work:

- the design of a multi-agents architecture supporting both scenarios, and the organisational top-down approach followed to identify the societies, the roles and the interactions of agents;
- the construction of the ontology O'CoMMA and the structuring of a corporate memory exploiting semantic Web technologies;
- the design and implementation of the sub-societies of agents dedicated to the management of the annotations and the ontology and of the protocols underlying these groups of agents, in particular techniques for distributing annotations and queries between the agents.

KEYWORDS:

distributed artificial intelligence, knowledge management, corporate memory, ontology, knowledge representation, multi-agent systems, semantic web, information retrieval.

INTELLIGENCE ARTIFICIELLE DISTRIBUÉE ET GESTION DES CONNAISSANCES : ONTOLOGIES ET SYSTÈMES MULTI-AGENTS POUR UN WEB SÉMANTIQUE ORGANISATIONNEL

RÉSUMÉ :

Ce travail considère les systèmes multi-agents pour la gestion d'un web sémantique d'entreprise basé sur une ontologie. Dans le projet CoMMA, je me suis focalisé sur deux scénarios d'application: l'assistance aux activités de veille technologique et l'aide à l'insertion d'un nouvel employé dans une organisation. Trois aspects ont été développés dans ce travail :

- la conception d'une architecture multi-agents assistant les deux scénarios, et l'approche organisationnelle descendante adoptée pour identifier les sociétés, les rôles et les interactions des agents ;
- la construction de l'ontologie O'CoMMA et la structuration de la mémoire organisationnelle en exploitant les technologies du Web sémantique ;
- la conception et l'implantation (a) des sous-sociétés d'agents chargées de la maintenance des annotations et de l'ontologie et (b) des protocoles supportant ces deux groupes d'agents, en particulier des techniques pour la distribution des annotations et des requêtes entre les agents.

MOTS-CLES:

intelligence artificielle distribuée, gestion de connaissances, mémoire d'entreprise, ontologie, représentation des connaissances, systèmes multi-agents, web sémantique, recherche d'informations.

Scientific Philosopher Doctorate Thesis In Informatics - Fabien GANDON - Thursday the 7th of November 2002 - INRIA and University of Nice - Sophia Antipolis - Doctoral School of Sciences and Technologies of Information and Communication (S.T.I.C.).