



HAL
open science

Support à la Divergence dans les Communautés de Partage de Connaissance

Alicia Diaz

► **To cite this version:**

Alicia Diaz. Support à la Divergence dans les Communautés de Partage de Connaissance. Human-Computer Interaction [cs.HC]. Université Henri Poincaré - Nancy I, 2005. English. NNT: . tel-00010910

HAL Id: tel-00010910

<https://theses.hal.science/tel-00010910>

Submitted on 8 Nov 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Supporting Divergences in Knowledge Sharing Communities

THÈSE

présentée et soutenue publiquement le 20 Octobre 2005

pour l'obtention du

Doctorat de l'université Henri Poincaré – Nancy 1

et

Doctorado en Ciencias Exactas - Universidad Nacional de La Plata

(spécialité informatique - especialidad informática)

par

Alicia Díaz

Jury

Président : Rose DIENG, Directeur de Recherche INRIA, France

Rapporteurs : Rose DIENG, Directeur de Recherche INRIA, France
Luis OLSINA, Professeur UNLPam, Argentina

Examineurs : Claude GODART Professeur UHP, ESSTIN, France
Gérôme CANALS, Maître Conférence Univ. Nancy 2, France
Silvia GORDILLO, Professeur UNLP, Argentina

Invité : Amedeo NAPOLI, Directeur de Recherche CNRS, France

Mis en page avec la classe thloria.

Abstract

Collaborative knowledge sharing systems (CKS systems) are groupware applications that allow on-line communities to share knowledge. These systems provide shared computational workspaces where the community's members can develop a shared knowledge repository (KR). Unfortunately, when comparing knowledge-sharing face-to-face or even lightweight groupware supported, CKS systems' activity seems forced, a little motivating and artificial. Because one of the problems with current knowledge repository systems is that they consider the knowledge-sharing activity as the centralized accumulation of information rather than as a process by means of which the group develops a common understanding (builds knowledge) and where knowledge divergence can take place.

In this research, I study knowledge-sharing communities, how they share knowledge and, in particular, I focus on the occurrence of knowledge divergence. Knowledge divergence means a cognitive conflict at the common understanding level and it is expressed through the generation of alternatives, arguments and different points of view about a conceptualization of a topic of interest. In this context, knowledge-sharing activity is seen as an evolutionary process and reflexes the discussion through which the participants reflect on a domain of interest, in order to build a shared conceptualization of their common understanding. In this conception of the knowledge-sharing activity, knowledge divergence occurrence is a common practice.

Assisting people in sharing knowledge and supporting their knowledge divergence can help to improve the conditions of usability of the CKS systems. In this dissertation, I hypothesized that any effort to reach a CKS system, as a groupware application that supports the knowledge-sharing activity, should pay special attention to the discussion activity and the knowledge divergence occurrences in order to improve the conditions of usability of the system. Knowledge-sharing activity puts forward special situations that will become the requirements to CSCW (Computer-Supported Collaborative Work) field and groupware systems. Therefore, the focus of this research is to find out a new CSCW approach that helps to carry out the knowledge-sharing activity with knowledge divergence occurrence.

I develop a theoretical framework which conceptualizes the knowledge-sharing activity and considers the occurrence of knowledge divergences as a first order element within it. This framework synthesizes the architecture of a CKS system as a groupware application which supports a knowledge-sharing process. This framework takes into account the collaboratively and distributed development of a shared KR (the conceptualization of its common understanding), the occurrence and coexistence of a knowledge divergences and the preservation of the autonomy of the participants. In particular, my goals are to show that an monotonic extension of the KR, differentiated workspaces for preserving individual autonomy, appropriately supports to manage knowledge divergence and suitable awareness services for keeping users aware of the knowledge-sharing activity can be framed in a sufficiently operational form to be useful to groupware designers and to improve the

conditions of usability of the traditional knowledge-sharing systems.

Keywords: knowledge-sharing communities, knowledge sharing, divergent knowledge, groupware, knowledge awareness.

Résumé

Les systèmes de partage collaboratif de connaissance sont des collecticiels qui permettent à des communautés en ligne de partager des connaissances. Ces systèmes offrent des espaces partagés où les membres d'une communauté peuvent construire ensemble une mémoire partagée. Malheureusement, ces systèmes restent peu utilisés car ils sont peu motivant, paraissent artificiels et le partage de connaissance est vécu comme forcé. Un des problèmes avec les systèmes actuels de partage de connaissance est qu'ils voient l'activité de partage comme une accumulation centralisée d'information plutôt que comme un processus évolutif dans lequel un groupe construit de la connaissance et dans lequel des divergences et des inconsistances dans la mémoire peuvent apparaître.

Dans ce travail, nous étudions les communautés de partage de connaissance, en portant une attention particulière sur la divergence. Cette divergence dans la connaissance mémorisée correspond à un conflit cognitif et se traduit par l'apparition d'alternatives, d'arguments et de points de vue différents autour d'un concept ou d'un sujet d'intérêt. Dans ce contexte, l'activité de partage de connaissance est comprise comme un processus évolutif englobant notamment des étapes de discussion où les participants expriment leurs points de vue, et dont l'objectif est la construction commune d'un concept partagé. Dans ce cadre, les divergence de point de vue sont des pratiques courantes.

En assistant les membres dans le partage et l'expression de leur divergences, nous pensons améliorer l'utilisabilité des systèmes de partage de connaissance. Dans cette thèse, nous faisons l'hypothèse qu'une attention spéciale dans le support de la divergence et des discussions autour de la divergence est une condition nécessaire pour concevoir et améliorer l'utilisabilité d'une application collaborative pour le partage de connaissance. Cette approche propose des besoins nouveaux aux applications de type TCAO/collecticiel. Ainsi, le thème central de ce travail est d'identifier et concevoir des mécanismes de gestion logicielle de la collaboration pour améliorer le support au partage de connaissance en supportant les divergences et inconsistances.

Nous développons un cadre théorique qui conceptualise l'activité de partage de connaissance et considère la divergence comme un objet de premier ordre. Ce cadre propose une vision d'un système de partage de connaissance comme étant basé sur une application collaborative conduite par un procédé de partage de connaissance. Il prend en compte le développement distribué et collaboratif d'une mémoire commune, l'apparition et la conservation de divergences et l'autonomie des participants. En particulier, nos objectifs sont de montrer qu'une approche de construction par augmentation monotone de la mémoire, des espaces de travail différenciés (privé/public) et des services de conscience de groupe adaptés afin de permettre aux utilisateurs de suivre l'évolution de la mémoire partagée, sont des éléments clés dans la conception d'un collecticiel de partage de connaissance.

Mots-clés: communautés de partage de connaissance, divergence, collecticiel, conscience de groupe, conscience de connaissance partagée

Resumen

Los sistemas para compartir conocimiento colaborativamente (Collaborative knowledge sharing systems —CKS systems) son aplicaciones de software colaborativos (groupware) que permiten a las comunidades "en línea" intercambiar conocimiento. Estos sistemas proveen de espacios de trabajo compartidos donde los miembros de la comunidad pueden desarrollar una memoria o repositorio de conocimiento compartido. Desafortunadamente, los sistemas CKS resultan forzados, poco motivados y artificiales cuando se los compara con como se comparte conocimiento presencialmente o incluso soportado con alguna herramienta groupware simple. Uno de los problemas de los actuales implementaciones de memorias de grupos es que ellas consideran la actividad de compartir conocimiento como la acumulación centralizada de información, en lugar de considerarla como un proceso a través del cuál se desarrolla un entendimiento común (se construye conocimiento), sino también donde pueden existir divergencias.

En la investigación comprendida en esta tesis, se estudiaron las comunidades que comparten conocimiento y como lo comparten, pero en particular, se focalizo en la posibilidad que ocurran divergencias. Por divergencia se entiende a la ocurrencia de un conflicto cognitivo en el entendimiento común y se expresa a través de la generación de alternativas, argumentaciones y diferente puntos de vistas sobre la conceptualización del tópico de interés de la comunidad. En este contexto, la actividad de compartir conocimiento se desarrolla a través de un proceso evolutivo que refleja la discusión en la cual la comunidad reflexiona sobre el dominio de interés, con el objeto de construir una conceptualización compartida del entendimiento común. Bajo este enfoque, la ocurrencia de divergencias es una práctica frecuente. Asistir a los miembros a compartir conocimiento y soportar sus divergencias puede ayudar a mejorar las condiciones de usabilidad de los sistemas CKS.

En esta tesis, se presenta como hipótesis que prestar especial atención a la ocurrencia de divergencias y a la actividad de discusión entrono a ella, redundará en la mejora de las condiciones de usabilidad de las aplicaciones groupware para compartir conocimiento. Este enfoque impone nuevos requerimientos en el campo de las aplicaciones de trabajo cooperativo soportado por computadora (CSCW) y de aplicaciones groupware. Es por esto, que el tema central de esta tesis es encontrar y concebir los mecanismos de gestión de software colaborativo que faciliten compartir conocimiento soportando la ocurrencia de divergencias.

Se desarrolla un marco teórico que conceptualiza la actividad de compartir conocimiento donde la ocurrencia de divergencia es considerada un objeto de primer orden. Este marco teórico sintetiza las características distintivas de un sistema CKS como una aplicación de software colaborativo que soporta el proceso de compartir conocimiento. El tiene en cuenta el desarrollo colaborativo y distribuido de una memoria de grupo, la ocurrencia y coexistencia de divergencias y la preservación de la autonomía de los participantes. En particular, esta investigación tiene como objeto mostrar que la extensión monotónica del repositorio de conocimiento, la diferenciación entre un espacio de trabajo (público/privado) y los servicios de awareness adecuados para seguir la evolución de la memoria compartida, son los elementos claves para la concepción de un software colaborativo para compartir conocimiento.

Palabras claves: comunidades que comparten conocimiento, divergencias, software colaborativo, awareness de conocimiento

Acknowledgment

There are a number of people to whom I wish to express my gratitude for their help during the period of my PhD and in particular, in the last period while I was writing this thesis.

Firstly, I wish to thank each and everyone of those who have given me the possibility to do this thesis. I would like to thank my colleagues of Lifia, who have motivated me to face this challenge at this moment of my life. But I am also grateful to Marie-Christine Imbert and Claude Godart for having offered me the opportunity to carry out this thesis in co-tutorship in France.

Secondly, I wish to thank each and everyone of those who have supervised this thesis. I would like to thank Claude Godart and Gustavo Rossi for having accepted the role of the advisors and given me the freedom to choose the subject of this thesis. But, I am particularly grateful to Gerome Canals, for having undertaken the difficult job of supervising the technical features of my PhD.

I would like to thank the authorities of the UNLP-Facultad de Ciencias Exactas and the UHP-ESSTIN for having created the legal framework to carry out this thesis. And I also thank the Loria's administration for having provided me with the place and the necessary services to work. This PhD has been funded by the Lifia lab, the INRIA, the UHP and the UNLP.

My gratitude goes also to Guillermo Baldo for making the effort to understand my ideas about the prototype, for his implementation and for having the enough patient when I suggested changes. I thank also the Groupfia group (at Lifia) for each comment about the prototype.

A special thanks to my partners of the Ecoo team who made me feel as I were at the Lifia. Thanks to who has share with me the office, the coffee breaks and lunches, and even the visit to "la Foire de Nancy".

I would wish also to thank to each and everyone of my partners of Lifia lab. A special thanks to the heads of the Lifia for providing me the time and the calm needed to finish this thesis. Thanks to Gustavo and Roxana for being in charge of my courses.

A particular thank you goes to my mother Rosa and my sisters, Daniela and Florentina, for always supporting me, for making it easier for me to take difficult decisions. Thanks to my father Reynaldo for being with me always and everywhere.

A special thank you to all those great friends who I have made at Nancy. Special thank to Françoise Clerc who has helped me a lot when I arrived to Nancy, has tried to teach me some French, has made me know "les brocantes de la Lorraine" and "la foret d'Haye". Thanks also to the Argentinean friends that I found in Nancy, Ariel, Ana and Carlos, who have opened their home to me. The last "thank you" is to my friends from La Plata, who have supported me during the development of this work.

to my family

Contents

1	Introduction	1
1.1	Knowledge-Divergence in Knowledge-Sharing Communities	2
1.2	Knowledge Management Approach to KS-Communities	4
1.3	CSCW Approach to Divergence Problem	6
1.3.1	Some Examples	9
1.3.2	Shared Workspaces and Group Awareness	11
1.4	Motivation and Research Hypothesis	15
1.5	Problem Statements and Contributions	16
1.6	Overview of the Dissertation	18
2	Knowledge-Sharing Activity Foundations	19
2.1	KS-Communities Foundations	20
2.1.1	Knowledge-Sharing as a Collaborative Activity	20
2.1.2	KS-Communities as Communities of Action	22
2.2	The Knowledge Repository Foundations	26
2.3	The Knowledge-Sharing Activity	32
2.3.1	A KS-Activity Scenario	33
2.3.2	The Community Knowledge	34
2.3.3	Knowledge-Sharing Process	36
2.4	Knowledge Divergence Occurrence	39
2.4.1	Cognitive Conflict	39
2.4.2	Reaction as a means for revealing discussion activity	41
2.5	CSCW Approach to Support KS-Activity	42
2.5.1	CSCW Requirements to Support KS-Activity	43
2.6	Conclusion	44

3	The Knowledge-Sharing Framework	47
3.1	Collaborative Knowledge-Sharing Framework	47
3.2	Knowledge Artifact	49
3.3	The Knowledge-Sharing Workspace	50
3.3.1	The Management of Knowledge Repository Versions	51
3.3.2	Private and Public Workspaces	53
3.4	Augmentative Development of the Knowledge Repository	55
3.5	Divergent Knowledge Management	58
3.5.1	Discussion Thread Model	59
3.6	Conclusion	63
4	Sharing Knowledge by means of Ontologies	65
4.1	Introduction	66
4.2	Knowledge Model: Ontological Artifact	67
4.3	Ontological Representation of the Knowledge Repository	70
4.3.1	Domain Ontology	70
4.3.2	Member Profile Ontology	71
4.3.3	Knowledge-Sharing Action Ontology	72
4.4	Sharing-Knowledge by means of Ontologies	72
4.5	Augmentative Ontological Contributions	75
4.5.1	Checking non occurrence of conceptual description mismatches	77
4.5.2	Ontologies Integration	81
4.6	The Occurrence of Ontological Divergences	81
4.6.1	Ontological Discussion Thread Components	82
4.7	Conclusion	84
5	Knowledge and Discussion Awareness	87
5.1	The Role of Group Awareness at the KS-Activity	88
5.2	Knowledge Awareness and Discussion Awareness	89
5.2.1	Knowledge Awareness	90
5.2.2	Discussion Awareness	93
5.3	Information Needs	94
5.3.1	Information Needs of Knowledge Awareness	96
5.3.2	Information Needs of Discussion Awareness	102
5.4	Knowledge Awareness Mechanism	109

5.4.1	Gathering of Knowledge Awareness Information	111
5.4.2	Delivering Knowledge Awareness Information	115
5.5	Conclusion	117
6	The Co-Protégé Prototype	119
6.1	Co-Protégé	119
6.2	Working in Co-Protégé	122
6.3	Co-Protégé Awareness	126
6.4	Co-Protégé Model, Metamodel and Generic Ontologies	127
7	Conclusion	135
7.1	Thesis summary	135
7.2	Results	140
7.3	Future Work and Research	141
	Appendixs	143
A	An Overview of Ontologies	143
A.1	Introduction	143
B	A Brief Introduction to Protégé	149
B.1	Introduction	149
	Bibliography	157

List of Figures

2.1	A schematic representation of the ks-activity carried out at scenario	38
2.2	The discussion thread of the scenario	43
3.1	A schema of the ks-workspace	51
3.2	KS-actions conceptual model	56
3.3	Discussion artefact.	60
3.4	Discussion thread conceptual model. It is an aggregations of discussion artefact.	61
3.5	Discussion action conceptual model	62
4.1	UML diagram representing the ontology knowledge model.	69
4.2	The conceptual model of an Ontological Knowledge Artifact.	69
4.3	Conceptual and concrete representations of the knowledge domain ontology corresponding to the scenario of section 2.3.1	71
4.4	The member profile ontology	71
4.5	KS-action ontology	72
4.6	Sharing Knowledge by means of Ontologies	74
4.7	A simple ontological artifact	77
4.8	Two-layers of an ontology	79
4.9	An example of a ontological discussion thread	83
4.10	The Conflict Ontology	84
5.1	Knowlege Awareness as a means of externalizing knowledge	92
5.2	The conceptual model of a low-level knowledge-awareness information item	112
6.1	A snapshot of Co-Protégé. Both private and shared ontologies can be appreciated simultaneously. The black rectangle remarks the associated property pane to the current ontology. In this example, property pane shows the properties of the Person class from the private ontology.	120
6.2	The Co-Protégé architecture.	121
6.3	A snap-shot of checking results of a contribution. In particular it is an aborted contribution because the checking has failed.	124
6.4	User tab.	125
6.5	Conflict tab.	125
6.6	Difference tab.	126
6.7	The visualization of the private divergence at the private side	127

List of Figures

6.8	The visualization of the collection of notifications	128
6.9	The Co-Protégé metamodel, model and generic ontologies.	129
6.10	The Co-Protégé metamodel, model and generic ontologies.	131
B.1	A snapshot of Protégé-2000	151
B.2	Protégé-2000's class :STANDARD-CLASS is detailed	153
B.3	The Protégé-2000 Project class	155

List of Tables

5.1. The Where question	97
5.2. The Who question	99
5.3. The What question	100
5.4. The How question	101
5.5. The When question	102
5.6. The Why question	103
5.7. The Where question for discussion awareness	105
5.8. The Who question for discussion awareness	106
5.9. The What question for discussion awareness	107
5.10. The How question for discussion awareness	108
5.11. The Why question for discussion awareness	109
5.12. The When question for discussion awareness	110
6.1. Publication operations	123

Chapter 1

Introduction

Contents

1.1	Knowledge-Divergence in Knowledge-Sharing Communities .	2
1.2	Knowledge Management Approach to KS-Communities . . .	4
1.3	CSCW Approach to Divergence Problem	6
1.3.1	Some Examples	9
1.3.2	Shared Workspaces and Group Awareness	11
1.4	Motivation and Research Hypothesis	15
1.5	Problem Statements and Contributions	16
1.6	Overview of the Dissertation	18

Collaborative knowledge sharing systems (CKS systems) are groupware applications that allow on-line communities to share knowledge. These systems provide shared computational workspaces where the community's members can develop a shared knowledge repository. Unfortunately, when comparing knowledge sharing face-to-face or even lightweight groupware supported, CKS systems' activity seems forced, a little motivating and artificial. Because one of the problems with current knowledge repository systems is that they consider the knowledge-sharing activity as the centralized accumulation of information than as a process by means of which the group builds knowledge (develops a common understanding) and where knowledge divergence can take place.

In this research, I study knowledge-sharing communities, how they share knowledge and, in particular, I focus on the occurrence of knowledge divergence. Knowledge divergence means a cognitive conflict at the common understanding level and it is expressed through the generation of alternatives, arguments and different points of view about a conceptualization of a topic of interest. In this context, knowledge-sharing activity is an evolutionary process and reflexes the discussion through which the participants reflect on

a domain of interest, in order to build a shared conceptualization of their common understanding. In this last conception of the knowledge-sharing activity, knowledge divergence occurrence is a common practice.

Assisting people in sharing knowledge and supporting their knowledge divergence can help to make CKS systems more usable. Knowledge-sharing activity puts forward special situations that will become the requirements to CSCW (Computer-Supported Collaborative Work) field and groupware systems. Therefore, the focus of this research is to find out a new CSCW approach that helps to carry out the knowledge-sharing activity with knowledge divergence occurrence. In particular, my goals are to show that differentiated workspaces for preserving individual autonomy, appropriately supports to manage knowledge divergence and suitable awareness mechanisms for keeping users aware of the activity can be framed in a sufficiently operational form to be useful to groupware designers, and to show that support to these requirements can improve the conditions of usability of the traditional knowledge-sharing systems.

The remainder of this chapter aims to set the problem of knowledge divergence occurrence in the scene of the knowledge sharing activity. First, it is introduced this problem. Secondly, the Knowledge Management's approach to of the problem of knowledge sharing is presented. Then, the issue of divergence is analyzed from the CSCW approach, some examples are presented, and it finishes by providing with a theoretical background of the shared workspace and group awareness concepts. This section, finally present the motivation, hypothesis, problem statements and contribution of this research.

1.1 Knowledge-Divergence in Knowledge-Sharing Communities

Knowledge-Sharing Communities (ks-communities) are groups of people sharing a common area of expertise and/or who search for solutions to common problems. In the context of this definition this kind of communities are also known as *Communities of Practice* ([Wenger98], [Wenger02]) or more precisely as *Communities of Action* [Zacklad03a]. A ks-community is thus not necessarily an authorized or identified group in an organization. People in a ks-community can perform the same job, collaborate on a shared task or work together on a product. What holds them together is a common sense of purpose and a real need to know what each other knows. Most organizations will hold several communities and most people belong to at least one of them [Brown95].

People find value in meeting this kind of communities because they typically share knowledge, they help each other to solve problems, they discuss and explore points of view and ideas or they simply develop a tacit and *common understanding*. They get bound by the value they find by learning together.

Due to this activity, ks-communities accumulate knowledge and develop its own perspective on their topic of interest as well as a body of common knowledge, practices, and approaches. This knowledge can be of many different nature; domain knowledge, social knowledge, activity knowledge. The knowledge-sharing activity (ks-activity) is the collaborative learning process by means of which a community accumulates knowledge

[Diaz04a]. This process is carried out in an iterative and incremental fashion through which the community knowledge is built [Stahl05a]; and as a consequence, its knowledge is constantly growing and evolving. Knowledge evolves thanks to participants' contributions; so that, community's members are the participants of knowledge evolution. Thus, each new knowledge contribution is a step forward in a new state in the knowledge building. A knowledge contribution is the fact of communicating publicly any knowledge to the community, and this means to bring knowledge from individual knowledge context to the community (or shared) knowledge context. The community knowledge context is the common understanding which a group of people develops during the knowledge sharing activity and represents the group's knowledge cognition [Stahl05a].

The ks-activity is characterized by the following features:

- it is carried out by a small group of people;
- It is a distributed activity; every member participates by bringing to the community what s/he knows;
- Members learn collaboratively, they need to exchange knowledge to develop a common understanding;
- Knowledge is constantly growing and evolving; while ks-activity is running, the community is active;
- A priori, there is no conflict of power, everybody can contribute with knowledge and nobody is the owner of the knowledge or the "truth";
- Members like preserving their individual autonomy; they make a difference between their individual and shared knowledge context;
- Cognitive conflicts may arise; as everybody can contribute with her/his, they can even contribute with divergent knowledge;
- People manage divergence, by discussing different perspectives and arguments
- People coexist with conflicts during they share knowledge.

Divergence means the generation of arguments and different points of view about a knowledge conceptualization. Divergence is generally considered as a conflict at the common understanding level. Despite the fact that this situation can be seen as unfavorable, it exactly describes how the agreed knowledge naturally emerges in a ks-community by the simple act of sharing knowledge. Although the achievement of a consensus may or not happen, the most interesting thing is the process that takes place while the community persists with a conflict. This process represents the *discussion* in which the participants are involved.

The most known way through which ks-communities share knowledge is by socialization. *Socialization* is the sharing of tacit knowledge between people. Knowledge does not become explicit and the organization as a whole cannot easily use it. Socialization is often the preferred way of learning and sharing between people. The assumption that knowledge is inseparable from the communities that create it, use it, and transform it, motivates the use of community concept in Knowledge Management (KM) field and different terms like communities of actions or communities of practice appears to define/understand/study them. On the other hand, *externalization* [Nonaka95] is an individual act by means of which people formulate the fundamentals of their own tacit knowledge in a way that can

be stored or formalized. This process turns tacit knowledge into explicit one. Many KM efforts focus on the externalization of knowledge, that is, in the conversion of tacit (personal/individual knowledge) into explicit (organizational/community knowledge). The externalization approach implies the development of a *knowledge repository* (KR), where knowledge is stored and/or formalized.

Knowledge divergence occurrences are very natural when knowledge is shared by socialization (both in real or virtual environment), because the more communication there is among people, the more opportunities there are for conflict [Easterbrook93]. On the other hand, naturalness is the cost when a formal approach is considered. Therefore, any technological approach that aims at supporting ks-activity must pay attention to it.

CKS systems are concerned with enabling people to develop ("design") collaboratively a KR, respecting a knowledge-sharing process. However, it is obvious that conflict may arise among people involved in a collaborative activity, and it is more obvious if a group is sharing knowledge by means of designing a shared KR. It is a fact, independently on if the conflict is inherent to the collaborative activity or if it causes problems to the collaboration. Particularly, in communities that share knowledge, conflicts are inherent to the ks-activity and differences among individuals' experiences, personalities and commitment make the potential to conflict occurrence. The understanding of conflict occurrence is part of the understanding of the collaborative activity. This must include an understanding of how collaboration may break down, and how collaborative work can continue even in the presence of conflict.

Although the problem of knowledge conflict has been well studied in the Artificial Intelligent (AI) field, this is not the focus of this thesis. I am not concerned with doing reasoning in inconsistent knowledge bases or multi-agent systems. On the other hand, I understand that the occurrence of knowledge conflict is inherent to the ks-activity, and the management and development of them facilitates the achievement of a common understanding. My approach is more related to the collaborative development of a common understanding, where knowledge conflicts are kept among the community members. As the ks-activity is inherently a collaborative activity, CSCW approach is more suitable to study the occurrence of conflict (as the occurrence of knowledge divergence) at ks-communities. The remainder of this section will be devoted to present the state of the art in both KM and CSCW field.

1.2 Knowledge Management Approach to KS-Communities

KS-communities have gained much attention in the field of Knowledge Management (KM). Traditionally, KM means the management of activities related to the identification, acquisition, preservation, spread and use of organization's knowledge to be able to respond to rapid changes in a knowledge-based economy.

KM field and, recently, the area of organizational learning have been in charge of the ks-activity since organizational knowledge is a key strategic resource. Over the past decade, there have been many research efforts from a variety of disciplines that have produced a considerable volume of literature on knowledge sharing [Davenport00], focusing on knowledge sharing in groups and organizations, reflecting the multiplicity of

perspectives from which knowledge sharing has been studied. Probably, the predominant perspectives focus on the realization of a competitive advantage through effective sharing of knowledge, where organizational knowledge is a key strategic resource and therefore an effective creation, storage, and application of knowledge which leads to a competitive advantage for the firm. However, other approaches focus on understanding the modes through which knowledge is transferred. Knowledge resides in people, artefact and procedures of the organization, whatever the organization may be (work-team, enterprise, on-line community), and this knowledge is transferred through "communication". One of the most discussed modes for knowledge sharing is organizational learning. Organizational learning is a process through which the knowledge held by individuals is amplified, internalized, and externalized as part of an organization's KR, [Nonaka95].

There are many approaches to develop KR systems, most of them following the traditional KM approach. These systems come from systems based on central repositories of knowledge which were built by knowledge engineers, to distributed systems which grant the users with full autonomy over knowledge exchange. The centralized paradigm views organizational cognition as a convergent process that collects peripheral "raw" knowledge from various sources and codifies it into a central repository [Bonifacio02b]. As a consequence, technology is viewed as a tool for enabling central control, standardization, high capacity, and robustness. In this approach, the organization managers, supported by knowledge engineers, collect and structure the contents of the organizational memory that will then spread, expecting employees to use it and update it. Most KM projects aim at creating large, homogeneous knowledge repositories, in which corporate knowledge is made explicit, collected, represented and organized, according to a single - shared - conceptual schema. Although, many efforts have been made at developing this approach (many business theories around commercial software products), KM systems are often deserted by users, because they often feel that the stored knowledge in the repository was detached from their real working practices or the work of constantly updating the KR was seen as extra work. Probably, users continue to produce and share knowledge as they did before, namely through structures of relations and processes that are quite different from those embedded within KM systems. Bonifacio et al. claims in [Bonifacio02a] that the cause of this disadvantage lies in the use of an inadequate epistemological model, which is coherent with a traditional paradigm of managerial control, but is in contradiction with the deep nature of knowledge.

In short, the main limitation of a centralized architecture is not technological, but organizational. This creates a mismatch between social form and technological architecture, and this often produces a rejection by users. In traditional KM approach, the knowledge divergence does not occur as part of the ks-community activity; it is not a part of the ks-activity. If it would occur, it is a concern with the knowledge engineers and it is solved before arriving at the implementation of the KR. However, recent developments show a shift in the focus of KM from knowledge organization to collaboration, becoming the focus of KM in the management and nurturing of collaboration among peoples ([Dignum03], [Bonifacio03]). This approach fits more appropriately to ks-communities and the occurrence of knowledge divergence. Therefore, CKS systems appear as an approach in this direction .

This recent approach involves KM and CSCW come together and, consequently,

new groupware applications arise to cover the KM needs ([Agostini03], [Birnholz03], [Poltrock03], [Cluts03], [Schrott03]). KM is a new domain to re-analyze CSCW hypothesis since it states a series of new requirements concerning the development of CKS systems. Particularly in this thesis, I focus on those requirements related to the occurrence of knowledge divergences in the cooperative development of a distributed KR. This approach moves us to re-think CSCW features such as workspace and awareness concepts, in order to fit them to a ks-activity with knowledge divergence occurrence.

1.3 CSCW Approach to Divergence Problem

The term *Computer Supported Cooperative Work* (CSCW) was first coined by Greif and Cashman in 1984, at a workshop attended by individuals interested in using technology to support people in their work [Grudin94]. According to Carstensen and Schmidt [Carstensen02], CSCW addresses "*how collaborative activities and their coordination can be supported by means of computer systems.*" On the one hand, many authors consider that CSCW and groupware are synonyms. Ellis, in [Ellis93], defines *groupware* as "*computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment*". On the other hand, different authors claim that while groupware refers to real computer-based systems, CSCW focuses on the study of tools and techniques of groupware as well as their psychological, social, and organizational effects. Wilson's [Wilson91] expresses the difference between these two concepts: *CSCW is a generic term, which combines the understanding of the way people work in groups with the enabling technologies of computer networking, and associated hardware, software, services and techniques.*

Key issues of CSCW are group awareness, multi-user interfaces, concurrency control, communication and coordination within the group, shared information space (workspace) and the support of a heterogeneous, open environment which integrates existing single-user applications.

Groupware, also known as *Collaborative software* [Ellis91], is software that integrates work on a single project by several concurrent users at separated workstations. Groupware technologies are typically categorized along two primary dimensions: whether users of the groupware are working together at the same time ("real-time" or "synchronous" groupware) or different times ("asynchronous" groupware), and whether users are working together in the same place ("co-located" or "face-to-face") or in different places ("non-located" or "distance") [Johansen88]. Groupware systems become more valuable when more people use it. For example, calendaring becomes more useful when more people are connected to the same electronic calendar and choose to keep their individual calendars up-to-date. Moreover, groupware is sometimes divided into three categories depending on the level of collaboration. They are communication tools, conferencing tools, and collaborative management tools. *Electronic communication tools* send messages, files, data, or documents among people and hence facilitate the sharing of information. (e-mail, faxing, voice mail, web publishing). *Electronic conferencing tools* also facilitate the sharing of information, but in a more interactive way (*data conferencing*: networked PCs share a common "whiteboard" that each user can modify; *voice conferencing*: telephones allow

users to interact; *video and audio conferencing*: networked PCs share video or audio signals; *discussion forums*: a virtual discussion platform to facilitate and manage online text messages; *chat rooms*: a virtual discussion platform to facilitate and manage real-time text messages; *electronic meeting systems*: a conferencing system built into a room). *Collaborative management tools* facilitate and manage group activities (*electronic calendars*: schedule events and automatically notify and remind group members; *project management systems*: schedule, track, and chart the steps in a project as it is being completed; *workflow systems*: collaborative management of tasks and documents within a knowledge-based business process; *KM systems*: collect, organize, manage, and share various forms of information).

An extension of groupware is *collaborative media*, software that allows several concurrent users to create and manage information in a website. According to the used method they can be classified in: web-based collaborative tools (such as Wiki and BSCW) and software collaborative tools (such as CVS or RCS). And, according to the domain of use they can be divided in: KM tools, knowledge creation tools and information sharing tools. This research, particularly, focuses on *knowledge creation* tools.

AS CSCW is concerned with empowering people to work together, it is natural that conflicts may arise among people in a collaborative activity, independently if the conflict is inherent to the collaborative activity or if it causes problems to carry out the collaborative work. The understanding of conflict occurrences is part of the understanding of the collaborative work. This must include an understanding of how collaboration may break down, and how collaborative work can continue even in the presence of conflict. *To assume absence of conflicts is naive* [Easterbrook93].

CSCW field studies conflict as a problem of articulation and coordination when people develop a computer-supported collaborative activity. Particularly, CSCW field has paid special attention to the problem of conflict occurrence because it is considered as a consequence of interaction. It is well known that *the more communication there is among people, the more opportunities there are for conflict* [Easterbrook93]. Potential conflicts in CSCW applications should be seen as an opportunity for interaction, and therefore should be most visible. Conflicts, where they exist, should be solved by the involved people [Borges00].

In the context of CSCW, as Dourish states in [Dourish95], the problem of divergences is considered as a consequence of working activities proceed in parallel (multiple streams of activity), during which time the participants are disconnected (divergence occurs); and periodically their individual efforts will be integrated (synchronization) in order to achieve a consistent state and progress the activity of the group. In this approach, the problem of divergence is considered more as a problem of synchronization and versioning.

However, in this research, the problem of divergences is a problem of coexistence with cognitive conflicts, where different community members has different points of view, and besides they would like to share their perspectives. This is the way to enable the community to develop its common understanding. The development of a decentralized KR is a design activity, where the community develops a common understanding, where knowledge divergences occur and discussion about the divergence should be promoted.

A CSCW application that supports ks-activity must consider the knowledge diver-

gence occurrence because it necessarily influences the cooperation style, it can change or reinforce relationships and patterns of interaction between collaborators. Conversely, if designers ignore issues of conflict in the explicit part of the design, then their underlying assumptions about conflict, or its absence, become embedded in the system. These assumptions may influence the style of cooperation leaving the community without the possibility to express knowledge divergence or to express it in unplanned ways. What is required, in these cases, is a KR which allows the knowledge divergence occurrence and the discussion interaction.

According to Easterbrook's classification of CSCW systems that support certain kind of conflict management [Easterbrook93], existing KR systems can be seen as information sharing tools, and/or concept development tools and/or collaborative writing environments. The two last correspond to knowledge creation tools. They are detailed below:

Information Sharing Tools are intended to help individuals in groups communicate with one another, and as such can be thought as computer-mediated communication (CMC) systems. However, while the emphasis in CMC is on the transport of information, information-sharing tools concentrate on the ways in which the meaning of the information can be more effectively communicated, and on the function of each message in the continuing dialogue among users. Hence, such systems have been designed to reduce the amount of misunderstanding caused by differing interpretations of messages. Weblogs and filtered emails are examples of these systems, that mainly support socialization as way of sharing knowledge, where knowledge remains hard-coded and conflicts can be observed tracking the posting of messages.

Concept Development Tools recognize conflict as a central component of the group's work, and in particular *the development (or design) of concepts*. The design process is regarded as "a dialectic between goals and possibilities" [Stefik87], with the goals and possibilities mutually inspiring one another while the concept is refined [Easterbrook93]. These systems are clear examples where divergence occurrence can be productive; conflicts at the design decision can be useful to achieve to a better result. It is possible for individuals to apply this technique, but it is more effective when used by groups. In group use, these systems can be also thought as information sharing tools. However, this distinction involves different emphases and attitudes towards conflict, because this means knowledge divergences and they occurrence and developing are considered as a source of knowledge. As the concept development activity is considered a wicked problem [Rittel73], and as the better way of tackling wicked problem is discussing them, most of these systems are argument-based systems. However, they emphasized more on modelling the discussion process, than on supporting accurately the development of a KR. Because most of them support the ks-activity as a socializing one.

Collaborative Writing Tool. One of the most popular application domains for CSCW is the support of collaborative writing (edition). This may be because it is an activity relevant to all researchers and designers. Writing complex documents is a design task, and therefore they can be seen as concept development tools, although the latter are more concerned with the elicitation of conflicts. The systems here also have similarities to those for information sharing. The emphasis in supporting collaborative writing has been on the manipulation and representation of the shared document, rather than on the communicative and linguistic aspects of the task. Here, divergence is understood as

the difference among versions of a document [Dourish95]. Many commercial applications concerning sequential documents (e.g., collections of reports, papers, and source code) can detect and show changes between new and old versions. There are two approaches: one shows differences between two versions of a document are visually displayed on the screen (sequential deltas, annotations and markups, highlighting, overviews, graphical playback,) and the other describes how document versions are kept or specified, and how changes within them are detected or tracked over time (file differencing, real time differencing, version control systems, history systems, configuration management, etc).

1.3.1 Some Examples

Currently, there are some CSCW approaches to support the ks-activity. Most of them suggest some kind of groupware applications. They go from simple information sharing tools to knowledge creation tools. However, they differ in the modality of sharing knowledge. While there are those that only support socialization, there are other that allows building repository of documents or the collaborative writing category. Although most of them are not co-located they can be either asynchronous or synchronous. These systems suggest the use of a virtual shared workspace to develop a centralized KR. Groups use such workspaces for collecting and structuring different kinds of information they need (e.g., messages, documents, graphics, spreadsheets, tables, or software) to achieve the goals of their collaboration. Among this kind of groupware tools, there are ones that support ks-activity and also take into account knowledge divergence issues.

The most known CMC systems are: emails, weblogs and instant messages. E-mails allow users to perform an asynchronous discussion where each contribution is a new email replying to a previous one. The KR can be seen as a temporal sequence of emails. Knowledge is hard-coded in the email's text and the knowledge divergence remain also hard-coded in the text, but also in the intention of the email. Some e-mailing tools support the discussion activity by chronological emails sorting, or by subject or sender and mixed with filter functionalities. Instant messages, may be, are the best tool to recreate a synchronous knowledge exchange. Knowledge documentation is achieved by storing the discussion, and, like emails knowledge is hard-coded in the text. Weblogs are a web application which contains periodic time-stamped posts on a common webpage. These posts are often but not necessarily in reverse chronological order. Such a website would typically be accessible to any Internet user. Collaborative Weblogs allow the development of forums. They are a good solution for asynchronous knowledge exchange, but similarly, with emails and instant messages applications, knowledge remains hard coded in the text of the postings. However, there are some groupware applications that improve CMC systems taken into account the discussion activity or the knowledge subject. For example, Webguide [Stahl99] or AulaNet [Gerosa01], propose computer-mediated communication (CMC) mechanism to support discussions through emails classification. This classification subsumes the dialogue components according to the domain of application. On the other hand, Gmail (Google mail), which allows grouping related messages, creates meaningful conversations. When one opens a message in a conversation, all messages will be stacked neatly on top of each other, like a deck of cards. As new replies arrive, the stack of cards grows. Grouping messages in this way allows one to quickly retrieve related messages and

see all your messages in context.

On the other hand, there are some KR systems that support ks-activity by giving functionalities for a collaborative management of a distributed document repository. For example, the *BSCW* (Basic Support for Cooperative Work [Appelt99]) system is based on the metaphor of shared workspaces where documents can be retrieved with ordinary web browsers. It is an information sharing tool. A further focus of the system is the information of the users about the activities within their workspaces, i.e., the system provides several awareness services about the activity and changes. Although the system primarily supports asynchronous modes of communication, it also provides some features for synchronous collaboration such presence awareness as well as interfaces to synchronous communication tools such as chat or audio/video conferencing.

Another, distributed KR application are *Wikis*. *WikiWiki or wiki* systems [Cunningham01] are websites (or other hypertext document collection) that allows any user to add content, as on an Internet forum, but also allows that content to be edited by any other user. It fits under collaborative writing category. The term can also refer to the collaborative software used to create a website. A wiki enables documents to be written collectively in a simple markup language using a web browser. One of the defining characteristics of wiki technology is the ease with which pages can be created and updated. Generally, there is no review before modifications are accepted, and most *wikis* are open to the public — or at least anyone who has access to the wiki server. In fact, even registration of a user account is not always required. It supports some kind of page versioning and a very few change awareness. Divergence can be expressed in this tools but it remains hard-coded in the wiki page text. The policy which regulates the discussion activity and knowledge divergence occurrences is responsibility of the participants and remains as implicit agreement among them.

On the other hand, *IBIS-based tools* are highly appropriate to manage knowledge divergence explicitly. Kunz & Rittel [Kunz70] developed the concept of an Issue-Based Information System with the purpose of developing a tool to support the coordination and planning of political decision processes. The IBIS is based on the principle that the design process for complex problems, which Rittel terms "wicked" problems, is fundamentally a conversation among the stakeholders (e.g. designers, customers, implementers, etc.) in which they bring their respective expertise and viewpoints to the resolution of design issues. *Ibis* model is the best structure to represent the discussion activity. The model imposes structural ('rhetorical') constraints on where and in what way classes of contributions may be added to the design. The whole IBIS paradigm is focused on the elicitation of alternative viewpoints ('positions'), a process that is clearly based on the assumption that the eliciting conflict is productive, and that the expression of the conflict in an objective form aids in its resolution. *Ibis*-based systems are very suitable to support knowledge divergence occurrence when knowledge conceptualization is considered as the object to be designed. The original model of rhetoric has been widely adapted and used to represent design argumentation. Many argumentation-based systems have been presented to support group discussion and design. Examples include ArgueTrack [Bouwer99], BetterBlether [McManus95], Belvedere [Suthers97], or other systems based on *Ibis* model [Kunz70], like G-*Ibis* [Conklin88] and currently Questmap [Conklin01]. Most of these systems were focus on structuring discourse with use of graphical interfaces to support

computer-based argumentation ([Yu99], [Buckingham94]). On the other hand educational systems, such as Webguide [Stahl99] or AulaNet [Gerosa01], improve computer-mediated communication (CMC) tools to support discussions and show how message categorization and structuring have energized discussions. In these systems, messages can be classified in more specific kinds in order to fix them to particular requirement of the each application domain ([Buckingham94], [Buckingham97]), for example in the learning environment AulaNet messages are classified according to the learning discourse. This classification allows AulaNet to type the emails.

All of them are worried about supporting discussion activity, however, as I have said above the knowledge remains hard-coded text as well the other kinds of tools. The knowledge is not machine-readable, and so there is still the need of a knowledge manager to represent in some paradigm.

Finally, semantic portals ([Maedche03], [Reynolds04]) can be introduced as an example where knowledge does not remain hard-code in the postings. For example, *OntoShare system* [Davies03] is an ontology-based WWW information sharing system that models the interests of each user in the form of a user profile and keep he/she aware of information changes according to the interest expressed in his/her profile. OntoShare has the capability to summarize and extract key words from WWW pages and other sources of information shared by a user and it then shares this information with other users in the community of practice whose profiles predict interest in the information. OntoShare is used to store, retrieve, summarize and inform other users about information considered valuable in some sense by an OntoShare user. OntoShare is base in an ontological approach to classify resources and understand user profile. OntoShare also modifies a user's profile based on their usage of the system, seeking to refine the profile in order to achieve to a better model of the interests of the user . Using of ontologies allows certain kind of knowledge formalization. Nevertheless, with respect to the collaborative ks-activity OntoShare only supports the development of a collaborative document repository, without given to the participants any possibility of interfering the ontology design or express different perspective in the document classification.

Every of this approaches make use of some kind of shared workspace which allows the development of the ks-activity and some kind of group awareness, however these concepts are dealt with a generic point of view, without paying special attention to the particularities of the ks-activity and knowledge divergence. In the following two subsections, I will unfold a brief generic overview of shared workspace and group awareness concepts are given. Readers that are familiar with these concepts may prefer to skip these subsections.

1.3.2 Shared Workspaces and Group Awareness

Any CSCW application can help the development of any collaborative activity by means of the definition of suitable shared workspaces, group awareness mechanisms to ensure coordination and articulation.

Shared Workspaces

In the real world, a shared workspace is a physical one where people can undertake some activity as a group. For example, a classroom is a workspace where teachers and students carry out the learning process. Workspaces can vary widely in their makeup: they can be small or large, two- or three-dimensional, connected or discontinuous. One of the reasons that people use workspaces is that they are a convenient container for task artefact –the visible and manipulable objects that are the focus of the activity. Artefact exist at both literal and representational levels. They are physical objects, and so can be manipulated in accordance with their physical structure. They are also markers for relevant concepts in a task, and so manipulations and relationships can often be interpreted in terms of the task.

This combination of space and artefact makes a shared workspace an *external representation* of the activity as a group ([Clark96] [Norman93] [Hutchins90]). Clark lists three uses of external representations in collaboration. External representations serve as a reminder of what is going on: "*the current state of the activity is represented in quite a concrete form*". External representations are useful for imagining possible moves or actions, with a consistent reference point to return to. They are also a means for task actions. That is, communication and interaction can be carried out by actions in the external representation. According to this last point shows that shared artefact and external representations can be used as means for communication in the shared workspace; however, other kinds of non-verbal communication are also supported. People can use gestures to demonstrate, can point to objects to identify them, and can confirm requests simply by carrying them out. These kinds of actions complement verbal communication and make it more efficient [Gutwin02]. To sum up, shared workspaces play a major role in the richness of interaction that we can see in collaborative situations. Any ks-activity uses a shared workspace that conceptualize the knowledge space, it is where the common understanding is developed through the manipulation of knowledge artefact.

Shared workspaces have natural constraints and affordances that shape the awareness that people maintain about one another. The most important property is that workspaces provide an environment for interaction, thus giving people something to be aware of. Three additional properties that affect awareness have been also recognized: perceptual availability, spatial organization, and bounded interpretation. By means of *perceptual availability* people can observe others as they move about the space and work on artifacts, they can see and recognize particular actions, they can see what tools others are using, and they can see where others are looking. *Spatial organization* refers to to specific locations in the workspace and artifacts are interpreted in part by their spatial location with respect to other sign. Furthermore, people often make use of spatial metaphors for organization. *Bounded interpretation* serve to provide a bounded environment that constrains interpretation, and allows people to map perceptual information.

There are many activities that people can undertake as a group. Group tasks range from conflict to cooperation, and involve both conceptual and behavioral activities [McGrath84]. Shared-workspace tasks are primarily *generation* and *execution*. Gutwin, in his thesis [Gutwin97], has compiled a set of basic tasks that a small group can do in a shared space. These categories are: construction, organization, dynamic control, creation and

design, and exploration. A *construction task* involves build, assemble, or compile a whole out of pieces, parts, or components. The shared workspaces becomes a place to put the structure and the pieces. The workspace artefact are the pieces and parts. There may also be tools, used to put the artefact together. Construction is an inherently spatial task. An *organization task* is in charge of the achievement of some state of organization in the artefact. Types of organization include sorting, ordering, arranging, categorizing, and scheduling. Although, involved artefact and relationships between artefact are more conceptual than physical, a spatial metaphor helps people to carry out an organization task. *Dynamic control tasks* are useful to keep a system in a certain state or control a by means of some procedure. This kind of task is characterized by the artefact can change or move over time autonomously. The workspace in a dynamic control task provides a space for observing the states of system artefact, and means for effecting controls on those artefact. The purpose of *design and creation* is to produce new entities that satisfy certain criteria. Writing, drawing, and brainstorming tasks, all involve the creation of new things. In these tasks, the artefact are the things created, and the workspace is where to put them. Artefact can either be representations (e.g. ideas in a brainstorming session) or literal objects (e.g. lines in a drawing). The goal of *exploration* is to find artefact in an environment that satisfy certain criteria. General examples of exploration include searching, hunting, gathering, and selecting. The artefact are determined by the problem domain, but the size of the collection is often the reason for the task being undertaken by a group. Example tasks include finding a set of sites on the world-wide web, or finding certain pieces of information in a loosely-structured information space.

A collaborative development of a KR may be considered both, a construction task and a design task. It is a construction task because the repository grows thanks to the accumulation of knowledge that is contributed by members; but it is also a design task because the conceptualization of this knowledge is achieved through brainstorming activity (discussion activity) where knowledge divergence may take place.

The type of task that a group undertakes also determines, to a certain extent, the kind of interactions that the group members will engage in, and therefore the kinds of awareness information they will need.

Group Awareness

In addition to explicit communication, in real world, groups benefit from implicit communication, such as indirect gestures, information about people's environment, or biographical information about people in a conversation. This information helps people to establish common ground, coordinate their activities, and helps to avoid surprises. Common ground is the mutual knowledge that people take advantage of to increase their communicative efficiency [Clark96]. Common ground is the basis to understand awareness concept. Previous researchers have defined awareness as "knowing what is going on" [Endsley95]; it is knowledge created through interaction between an agent and its environment. This conception of awareness involves states of knowledge as well as dynamic processes of perception and action. Four basic characteristics run through prior work on awareness [Gutwin02]: awareness is knowledge about the state of an environment bounded in time and space; environments change over time, so awareness is knowledge

that must be maintained and kept up-to-date; people interact with and explore the environment, and the maintenance of awareness is accomplished through this interaction; and finally, awareness is a secondary goal in the task; the overall goal is not simply to maintain awareness but to complete some task in the environment.

The term *awareness* (as Schmidt has established in [Schmidt02]) refers to *actors' taking heed of the context of their joint effort*, but may be, the widest spread definition is that given by Dourish and Bellotti in [Dourish92] where *awareness is the understanding of the activities of others, which provides a context for your own activity*. This is a very general definition that the different authors refine to give more specificity classifying them with different adjectives. Schmidt in [Schmidt02] has enumerated them as: general awareness, collaboration awareness, peripheral awareness, background awareness, passive awareness, reciprocal awareness, mutual awareness, workspace awareness, etc. This list should be completed with the recently appeared "change awareness" [Tam04]. Next, I will briefly detail workspace and change awareness because they are the more useful to ks-activity.

Workspace awareness is a kind of awareness that has an intimate relationship with shared workspaces. Workspace awareness involves knowledge about: where others are working, what others are doing, and what they are going to do next. This information is useful for many of the activities of collaboration: for coordinating action, managing coupling, talking about the task, anticipating others' actions, and finding opportunities to assist one another [Gutwin97]. Workspace awareness is defined as the *up-to-the-moment understanding of another person's interaction with the shared workspace*. It is limited to events happening in the workspace. This means that workspace awareness differs from informal awareness of who is around and available for collaboration, and from awareness of cues and turns in verbal conversation, both of which have been studied previously in CSCW ([Dourish92], [Greenberg96]). Workspace awareness is made up of many kinds of knowledge, they give a basic idea of what information to capture and distribute in a groupware system. The basic set is the elements that answer "who, what, where, when, and how" questions. That is, when we work with others in a physical shared space, we know who we are working with, what they are doing, where they are working, when various events happen, and how those events occur. People keep track of these things in all kinds of collaborative work, and these are the kinds of information that should be considered first by designers. Within these basic categories identify specific elements of knowledge that make up the core of workspace awareness. Workspace awareness applies to both synchronous and asynchronous situations.

On the other hand, *Change Awareness* (or asynchronous change awareness of artefact in large) is *the ability of individuals to track the asynchronous changes made to a collaborative document or surface by other participants* [Tam04]. In same-time collaborations, people use workspace awareness not only to follow actions of others, but to understand and respond to any changes others make to the workspace artifact. But when people interact asynchronously this awareness disappears; changes are only understood if one person tells the other what they have done, or if the person can understand the changes made by inspecting the artifact. If people cannot understand what has changed, collaboration can quickly move out of control. To be aware of the changes a user needs to know whether there is something different since s/he last looked at the work. It is a simple and general question, but it will adapted to the task that is being performed, the

person who is carrying out the task, as well as the surrounding environment. Following Gutwin's approach [Gutwin02], Tam has described at a high level the questions that may be asked. This set of questions includes: *where, who, what, how, when, why* changes have taken place. Besides, this kind of group awareness takes into account for the fact that people may need to view from different perspectives. In particular, a person may query the workspace for changes from an artifact-based perspective, from a person-based perspective, or from workspace-based perspective. Of course, there is a strong relation between the three workspace perspectives and the six categories of awareness questions, for example, an individual that holds a person-based perspective will heavily focus on the 'who' category of questions, whereas someone that holds an artifact-based view of the workspace may focus instead on the 'what' category of questions and try to determine what changes were made to specific objects. The main point is that the person's particular view of the workspace will influence the value that he or she attaches to each category of question.

In the context of this research, although, the workspace awareness and the change awareness may be considered as the kinds of awareness that would be useful to the collaborative development of a shared KR; they have to be improved to take advantage of the particular subject of collaboration. When the collaborative activity consist of the development of the common understanding of a group, group awareness plays a new role in the development of the ks-activity, I expect that awareness, or knowledge and discussion awareness as I have called them, becomes more appropriated to this kind of activity and take advantage of the involved artefact —the knowledge and the knowledge divergence.

1.4 Motivation and Research Hypothesis

The main motivation of this research is to assist ks-communities to share knowledge with the occurrence of knowledge divergence. KS-communities exchange and discuss about a subject of interest as part of a collaborative learning process by means of which they develop a common understanding. Despite the fact that knowledge divergence occurrences can be seen as unfavorable, it exactly describes the process through which the agreed knowledge naturally emerges in the community by the simple act of sharing knowledge. This process represents the *discussion* in which the participants are involved and through which they can express their points of view of their conceptualization of the domain of interest.

The field of KM has paid a special attention to this kind of communities, because it see them as the source of the knowledge of the organization. KM has proposed many supports to the ks-activity. Recently, decentralized KR systems emerges as a better approach which focuses in the management and nurturing of collaboration among peoples. Decentralized KR systems are KR systems which are developed collaboratively. In this address, Collaborative Knowledge Repository (CKR) systems, as a groupware applications, appear as an particular kind of decentralized KR systems. CKR systems makes that KM field meets CSCW field, understanding a decentralized KR systems as a groupware application where the KR is developed collaboratively. However, although there are many effort in this address, CKR systems seems to be forced, few motivating and artificial in comparison

with face-to-face ks-activity.

This usability problem is caused in part because current CKR systems are considered as systems to collaboratively accumulate knowledge and thus to develop the KR. But they still leave outside the possibility of supporting the ks-activity such as it is. Although, it is useful for a ks-community to collaboratively develop a KR that accumulates its own knowledge, it also needs to be assisted to carry out the ks-activity as a means to build this knowledge. And in this address, it is important to consider the ks-activity as a whole, where people do not only contribute with knowledge, but they also exchange knowledge divergences as another way of exchange knowledge.

In this thesis, I suggest that any effort to reach a collaborative knowledge sharing (CKS) system, as a groupware application that supports the ks-activity, should pay special attention to the discussion activity and the knowledge divergence occurrences to improve the conditions of usability of the system. This system should be characterized by: the collaborative developing of a KR, by means of which each participant contributes to the development of the conceptualization of the common understanding; and the occurrence and coexistence of knowledge divergence, as part of this collaborative activity.

1.5 Problem Statements and Contributions

I will investigate and test previous hypothesis by the conceptualization of a collaborative knowledge-sharing framework that consider the occurrence of knowledge divergences as a first order element within the ks-activity. This framework has to conceptualize the architecture of a CKS system as a groupware application which supports a knowledge-sharing process. This process has suitably to describe the ks-activity. This frameworks has to take into account the collaboratively and distributed development of a shared KR (the conceptualization of its common understanding), the occurrence and coexistence of a knowledge divergences and the preservation of the autonomy of the participants.

To described this framework I will conceptualize the knowledge-sharing process and the shared knowledge in order to find out the requirements to be taken into account to build this framework, which will synthesize the concepts of a knowledge-sharing workspace, knowledge divergent management and knowledge and discussion awareness; and finally will evaluate the viability of the resulting framework in a prototypical system. These goals will be described by following objectives:

Objective 1. I will conceptualize the ks-activity and the shared knowledge in order to find out the CSCW requirements to suitably support this activity. This conceptualization will be met by: given a conceptualization of the ks-activity as the process by means of which knowledge is converted from tacit to explicit knowledge and goes from private to public knowledge context and it also may be divergent. The other goal is to conceptualize the different nature of the knowledge that is shared by the community (domain knowledge, social knowledge, activity knowledge). This objective will be successful if it allows me to find out the CSCW requirements to be taken into account to build this framework.

Objective 2. I will build a conceptual CKS framework that synthesizes the requirements of a groupware application to support the ks-activity. This framework states the concepts

of a knowledge-sharing workspace, knowledge representation formalism and knowledge awareness and discussion awareness. This framework has to support the necessary functionalities to cover the ks-activities expressed in terms of the KR and the knowledge-sharing process—externalization, publication internalization and reaction activities. This CKS-framework has to distribute these functionalities in the next components:

— A *Knowledge-sharing workspace*. A shared workspace that facilitates the ks-activity should preserve the characteristics of the ks-activity by supporting the externalization of the knowledge through some knowledge representation formalism; by respecting private and shared knowledge context; by facilitating any knowledge contribution, even they were contributions of divergent knowledge;

— The *Representation of private and shared knowledge context*. People need to differentiate between private and shared knowledge context. Knowledge internalization is a private activity, whereas the submission of a contribution is a public activity.

— A *Knowledge representation formalism*. For externalization, it is mandatory to count with a mechanism that allows the ks-community to make a conceptualization of its common understanding. However, the CKS-frameworks has to be independent from it.

Knowledge Divergence Management. Having an appropriated representation of the discussion. The discussion thread is the artifact through which members can express cognitive conflicts.

Knowledge and Discussion Awareness. People need to be aware of the ks-activity and mainly of the occurrence of divergences. Otherwise, the support of divergence loses its worth.

The suggested conceptual framework will be successful if it can organize the design of the knowledge-sharing workspace, which support knowledge representation, private and public workspace, facilitates the development of the knowledge-sharing process and discussion thread management and provides suitable knowledge and discussion awareness services.

Objective 3. I will show the viability of the development of a CKS system as an instance of the previous conceptual framework. This objective will be met by two works. One is by instantiating the CKS-framework by the ontological paradigm as the knowledge representation system. The other is by the development of a prototypical collaborative KR where knowledge sharing is understood as the collaborative development of a shared ontology. I will consider that this objective has been met successfully if the system is reliable, valid and shows significant results.

This research will contribute with original ideas, knowledge and practice in the domain of knowledge-sharing where knowledge conflicts are considered as a first order elements within this collaborative activity. There are three main contributions:

- I will identify and describe the requirements of a CKS system to support ks-activity with knowledge divergence occurrence. Although CSCW field has previously identified these requirements, they have not been applied and tested before in the domain of ks-communities, even when conflict is recognized at knowledge understanding level as an inherent requirement of this domain.
- I will construct a conceptual framework that synthesizes these requirements to

support the ks-activity. This framework states the concepts of knowledge-sharing workspace, knowledge divergence management and knowledge awareness and discussion awareness.

- I will develop a prototypical application which allows the development of a shared ontology as if this activity was a knowledge sharing activity with knowledge divergence occurrences.

1.6 Overview of the Dissertation

The remainder of this dissertation is organized in seven chapters as follow. To set the scene, Chapter 2 provides a background on the fundamental elements of this research—ks-communities, ks-activity with knowledge divergence occurrence. This chapter is also focused on conceptualizing the knowledge-sharing process and integrating the occurrence of knowledge-divergence as part of it. Finally, the requirements to design a CKS frameworks as a CSCW application, which supports the previous knowledge-sharing process, are enunciated.

Once this foundation is presented, the second part deals with the conceptualization of the framework begins with the Chapter 3. The CKS framework describes the fundamental components of groupware application that implements a CKS system. These components are: the knowledge-sharing workspace, the divergence management component and knowledge and discussion awareness services. In this chapter, the knowledge-sharing workspace and the knowledge divergence manager will be introduced.

The ks-frameworks presented in Chapter 3, then, will be reconsidered in Chapter 4 to instantiate it by a particular knowledge representation system. Ontologies paradigm was chosen to represent the knowledge.

Chapter 5 presents the two group awareness services which are used to improve ks-activity: knowledge and discussion awareness. It introduces an awareness framework to understand the needed awareness to support the ks-activity.

Chapter 6 is about the prototype. It is an instantiation of the CKS-framework by ontology-based KR. This prototype is an extension of Protégé software.

Chapter 7, this chapter summarizes the main findings and contributions of the research, and suggests a number of directions for further study.

Chapter 2

Knowledge-Sharing Activity Foundations

Contents

2.1	KS-Communities Foundations	20
2.1.1	Knowledge-Sharing as a Collaborative Activity	20
2.1.2	KS-Communities as Communities of Action	22
2.2	The Knowledge Repository Foundations	26
2.3	The Knowledge-Sharing Activity	32
2.3.1	A KS-Activity Scenario	33
2.3.2	The Community Knowledge	34
2.3.3	Knowledge-Sharing Process	36
2.4	Knowledge Divergence Occurrence	39
2.4.1	Cognitive Conflict	39
2.4.2	Reaction as a means for revealing discussion activity	41
2.5	CSCW Approach to Support KS-Activity	42
2.5.1	CSCW Requirements to Support KS-Activity	43
2.6	Conclusion	44

Knowledge sharing is the best added value that people find when they meet to communities. Any technological approach for supporting on-line communities should then offer mechanisms to support it. To achieve this goal, significant attention must be paid to the collaborative process that allows communities to share what they know in a coherent way throughout their activity.

First, in this Chapter, the theoretical background of ks-communities is introduced. In this section ks-communities are studied as "communities of actions". Then, the KR

foundations are developed, where it is paid special attention to the distributed KR development. Next, it is presented the knowledge sharing activity as a collaborative process through which the community shares knowledge and builds its own KR. This spiral process consists of four steps: externalization, publication, internalization and reaction. In the context of this process I analyze how conflicts appear (divergence occurrences) as a consequence of reaction step. Finally, I identify the requirements of a CKS system that allow the community to follow the knowledge sharing process and support divergence occurrences.

2.1 KS-Communities Foundations

This section will be in charge of providing a theoretical background to understand ks-communities as communities of actions. First, knowledge-sharing activity will be understood as a collaborative activity from the CSCW point of view. Then, ks-communities are studied from different approaches such as situated action, communities of practice distributed cognition and coordination mechanisms, to, finally, conclude to ks-communities fit in "communities of action" because although they can be considered as communities of practice, to make knowledge explicit is a relevant feature of them.

2.1.1 Knowledge-Sharing as a Collaborative Activity

First of all, a clear idea of what a collective activity is should be introduced. Maybe the definition of K. Schmidt and C. Simone is the best known; according to them *cooperative work is constituted by the interdependence of multiple actors who, in their individual activities, in changing the state of their individual field of work, also change the state of the field of work of others and who thus interact through changing the state of a common field of work* [Schmidt96]. However, this is a general definition that also applies to groups where members are aware neither of the common goals nor of the activity of their partners, as it occurs, for example, among the users of an airplane reservation system cooperating via the system. Even, the centralized KM approach fits in this case, because although organization members are able to access to the shared KR, they may not share the goal of the KR or even do not know the activity of other organization members. Understanding ks-activity from this point of view is one of the reasons to get to problems of usability, as they were mentioned above. People, who work on a centralized KR do not feel neither motivated or engaged in the process of building collaboratively the KR. Besides, this approach does not consider discussion of contents as an inherent task of the ks-activity.

On the other hand, Zacklad is more worried about those collective activities in which the actors are aware of both the goals pursued and the deployed means to achieve them, and in which the characteristics of the organization and those of the environments are not to be strictly standardized. He proposes communities of action framework as a tool designed for the analysis of small groups working to achieve goals and to assist the design of distributed computer environments developed for this purpose. According to Zacklad, *"cooperative activities are collective activities oriented towards goals in which the means of designing and attaining the goals are neither completely formalized nor standardized.*

The actors therefore have a significant amount of autonomy and are free to define their modalities for coordinating their contribution and adapt themselves to emergent situations" [Zacklad03a].

To describe cooperative activities, Zacklad has developed the "*intellectual transactions theory*" focusing mainly on the role of linguistic communications in the analysis of collective (group, community) activities ([Zacklad03a], [Zacklad03b]). This theory has been originated in three disciplines: cognitive science, organizational and management science, and in social psychology of interactions. It states that interactions occur between "cognitively interdependent" actors and can be described as "transactions" corresponding to "a sharing of personal knowledge and to a reciprocal commitment". These interactions are therefore primarily viewed as an exchange, a sharing or a "mutual gift" not mainly involving objects but knowledge and confidence.

The use of the various kinds of knowledge in intellectual transactions (epistemic or relational) allows making the distinction between different levels of coordination in organized actions. These coordination levels are:

- *Mutual perception*: directly collection and mutually exchange of information (about the activities of the partners and their fields of operation) carry out the coordination.
- *The standardization of the knowledge or the relationship*: the intellectual transactions, on which the coordination is based, are integrated into a routine program that assigns predefined roles to the actors and breaks down the field of operation and the procedure into normalized objects.
- *The abstraction of the knowledge or the relationships*: intellectual transactions refer to the "principles and reasons" justifying the transactions, which can be based on either technological or scientific knowledge about the field of operation or organizational knowledge relating to the actors and the group as a whole.

Groups can develop two kinds of social relationships: *community* or *associative*. A community relationship "*relies on the subjective feeling of the two parties to mutually belong to each other and to be fully committed in the existence of the other*"; on the other hand an associative relationships "*result from a will or rational and interested motives rather than an affective identification*" [Weber19]. According to this, Communities of Practice [Wenger98] and Communities of Action fall into the category of community relationship, because they can be characterized by long-term collective activities, the development of a common language, and mutual learning in the course of action. However, they also possess some characteristics which are typical of the associative social relationship, such as the "voluntary" nature of the association between the members and the importance of defining "common goals" to address the collective activity, which make this activity more "rational" [Zacklad03b].

Ks-activity, in small groups, fits to this last definition of collaborative activity, because it needs less level of standardization and formalization of the activity in order to achieve a better ks-activity. This approach is more appropriate to manage situation of knowledge divergence occurrence where its evolution is uncertain. This approach also guarantee the principles of autonomy and coordination that are required to support a distributed-KM approach (see section 2.2.3).

2.1.2 KS-Communities as Communities of Action

Following, it is introduced a theoretical framework to contextualize ks-communities as *communities of action* [Zacklad03a], instead of another kind of community such as *communities of practice* [Wenger02]. Besides, I even compare them to coordination mechanism and articulation approaches that give the conceptual theory of CSCW systems design. First, I begin by introducing the situated action paradigm and communities of practice approach, to then explain the problems of these two approaches to model ks-communities. Next, I will do the same but from the CSCW approach, considering coordination mechanisms and articulation approach.

Some authors, like Lorenz [Lorenz01], claim that there exists a close relationship between situated action and communities of practice approaches; because in these two approaches, the problem solving abilities of the individuals emerge from concrete, situated practices, while the material and social environment constitute the essential resources for the orientation of action and the knowledge mainly remains tacit and contextualized.

Situated Action understands that there is a close dependency between the knowledge and the context where the action takes place, being the action environment a determinant resource for the management of the cognitive process. Spatio-orientational arrangements are crucial to achieving the "shared focus of attention" [Suchman87] which requires problem-solving under conditions of specialization and the labor division. Problem-solving skills emerge out of people's actual practice in a particular context and the knowledge remains both highly contextualized and tacit. It is precisely this highly contextualized and tacit nature of their knowledge, that makes the situated learning approach difficult to apply to the field of ks-communities behavior.

Communities of Practice approach has gained a particular interest in the field of Knowledge Management [Wenger98] because they are considered as the source of knowledge and the notion of *communities of practice* was developed, at least in part, as an effort to link organizational structure to organizational knowledge and problem solving. Communities of practice (CoP) consist of people bound together by informal relations who share a common practice. Around this shared practice they develop a common language and shared (common) understandings of the environmental context in which they work, including the meanings they attach to the manipulated artefact. Researchers such as Brown and Duguid ([Brown91], [Brown95]) and Wenger [Wenger02] maintain that firms consist of collections of possibly overlapping communities of practice and that these communities are central to both the transmission of knowledge within organizations and the organization's problem-solving capabilities. As Wenger recently put it, "*communities of practice are a company's most versatile and dynamic knowledge resource and form the basis of an organization's ability to know and learn.*"

On the other hand, the theories of distributed cognition and coordination and articulation, which frames the CSCW systems design, may be another approach to be taken in order to understand ks-communities.

Distributed Cognition. The most interesting contributions in the theories of distribution cognition have been done by the work of Hutchins [Hutchins95]. Although, Hutchins agree with previous approaches in that "*knowledge is necessarily situated relative to a context*", what it difference is his emphasis on the cultural and historical determinants of cognitive processes; and it is particularly due to the use of instruments. Hutchins's work also shows that in strongly instrumented environments, the used of artefact among the members of a group allows them to adapt to complex situations and to overcome to an incident without the need of complex dialogues.

Coordination mechanisms. The coordination mechanisms presented by Schmidt and Simone in [Schmidt96] are rather close to previous theories and aim to provide the conceptual basis to the CSCW systems design. Based on the definition of collaborative activity given previously in this section, the authors identify a specific task of coordination that is the *articulation* as the need of restraint the distributed nature of complex interdependent activities. According to them, it is a recursive activity because the management of an established arrangement of articulating a cooperative effort may itself be conducted as a cooperative effort which, in turn, may also be articulated.

Articulation work is made up of two components. Schmidt and Simone defined a coordination mechanism as a specific organizational construction, consisting of a coordinative protocol (an integrated set of procedures and stipulating the articulation of interdependent distributed activities) and an artifact (a permanent symbolic construct) in which the protocol is objectified. Coordination protocol is a resource for situated actions. It reduces the complexity of articulating cooperative work by providing a pre-computation of task's interdependencies. Thus, actors, for all practical purposes, can rely on to reduce the space possibilities by identifying a valid and yet limited set of options for coordinative action in any given situation. While artefact are fundamentals in the coordination mechanism to objectify and give permanence to the coordinative protocol so that its stipulations are unceasingly publicly accessible. The artefact represent dynamically, to certain level of granularity, the state of execution of the protocols.

The theories of distributed cognition and coordination mechanisms seem to be fairly complementary. The first of these theories focuses mainly on the role of the *instruments* used in the coordination of collective action, and the second, on the role of specific *artefact*, the "coordination mechanisms", which materialize the results of "articulation work".

Recently, Communities of Action emerge as a better framework to study ks-communities.

Community of Action Zacklad has defined "community of action" (CA) "*to avoid the traditional opposition between community and associative relationship when dealing with small groups which actively and thus to some extent rationally pursue explicit goals while relying on a tightly woven fabric of relationships to promote mutual sympathy and the mimetic learning that is assumed to characterize primary groups and communities of practice*".

Although the continuous activity of CA certainly allows their members to benefit from partly tacit shared knowledge, Zacklad also conclude that these communities are also *intended to develop an explicit, systematic body of knowledge which can be used to extract the know-how and the informal relations required to be able to recurrently redefine both the nature of the services they are supposed to provide and the internal organization on which they rely in the context of structurally open cooperative situations.*

A CA is characterized by: the duality of goals and the duality of knowledge. *Duality of goal* means that a CA has to work towards two kinds of goals simultaneously: transforming an external situation, and constructing an internal social surrounding allowing its members to develop mutual knowledge and identities while enjoying mutual sympathy (this does not avoid the occurrence of confrontations which can be animated but are carefully regulated). The goals of the first kind, called "services goals", tend to be reached as the result of epistemic intellectual transactions between the members, whereas those of the second kind, that we call "integration goals", rely on relational transactions. Through the consideration of duality of goals, CA recognizes the possibility of arriving at conflictive situations, such as knowledge divergence occurrence and even through the services goals, it gives the framework to fit the development of a "discussion around the divergence". On the other hand, *duality of knowledge* refers to the fact that collective *knowledge is not viewed as being only of a tacit, local and situated nature.* General knowledge and the construction and use of principles and laws based on hypothetical and deductive reasoning processes are also part of the activity of these communities. By means of duality of knowledge, the existence of explicit knowledge is possible, given the necessary fundamentals to the development of a shared KR where knowledge is explicit,

Depending on the types of goals and knowledge types, collective activities are classified as:

- *operational activities* correspond to the performance of the concrete tasks involved in achieving services goals. These goals have to do with the environment of the CA, which they transform by producing information, services or required objects, for instance.
- *strategic activities*, which also focus on the environment, however they are more abstract and point to define external goals, like delimiting the environment, defining "customers" and partners and appropriate principles on which to base operational activities and to plan their future development.
- *relational activities* are required to construct the social network and the individual identities within the community of action in a situated context. They fulfil the needs and reach the goals of integration which focus on the actors themselves and promote mutual discovery.
- *integrative activities* are those involved in constructing the organization of the community of action, its legitimate actors, the internal procedures of debate and decision-making, and the principles on which these are based.

A *CA* can be differentiated depending on the degree of specialization of their members in the dimension of the "services goals" or of the "integration goals". Specialization, both because it is acquired via organizational or professional knowledge of the field or because it corresponds to a specific innovation, is a prerequisite for solving problems and achieving goals. Communities of action try to maximize their efficiency by expressing different opinions about the problem in consideration and by organizing debates at which these are confronted and compared. The expression of different viewpoints generally reflects the divergent interests of the members, which are not of course abolished by the common goals that bring them together. But Zacklad hypothesizes that appealing to specialization can also be a strategy to generate specific competencies. Therefore, it is necessary to organize a confrontation to integrate complementary and divergent viewpoints and to construct the solution to the problem.

CA theory is more suitable to situate ks-communities because:

- CA keeps the properties of CoP, because they are the source of knowledge. In both approaches, situated cognition paradigm and communities of practice, the problem solving abilities of the individuals emerge from concrete, situated practices, while the material and social environment constitute the essential resources for the orientation of action and the knowledge mainly remains tacit and contextualized. The theories of situated action and communities of practice are more open to learning situations, and they give the linguistic interactions between actors an important role, although little attention is paid to the details of the mechanisms underlying these interactions.
- CA allows the existence of explicit knowledge. By considering duality of knowledge, it is possible to guarantee the existence of explicit knowledge, given the necessary fundamentals to the development of a shared KR. However, in CoP the shared understandings remain mostly tacit or implicit, and communities of practice are characterized by a low degree of codification of members' knowledge. As Brown and Wenger have underlined, the knowledge of communities of practice is tacit, weak of formalizing and it is transmitted as consequence of socialization, for example through storytelling.
- CA tries to maximize its efficiency by allowing expressing different opinions and by organizing discussions. CA theory attribute a leading role to the linguistic interactions occurring between the actors or to the profound impact these interactions can have on the on-going design of working situations at the organizational and cognitive levels, mainly when confrontation and divergent viewpoints are considered essential to the ks-activity. However, distributed cognition, and coordination and articulation theories does not attribute a leading role to the linguistic interactions.

Therefore, while CA theory allows us to situate ks-communities, the CSCW theories (distributed cognition and coordination and articulation theories) are highly relevant to take a distributed approach of the development of the KR and to support the ks-activity.

2.2 The Knowledge Repository Foundations

The aim of this section is to introduce the foundations of the collaborative development of a KR. First, it is given basic definitions of data, information and knowledge and an identification, categorization and classification of knowledge assets within a given organization. Then, the knowledge sharing foundations are explained from the traditional point of view of KM, where the community is considered as the source of knowledge. As the development of a KR is the traditional approach to support ks-activity; centralized and distributed approaches are also discussed.

2.2.1 Knowledge Classification

In a knowledge sharing management initiative, a first step should be an effective identification, categorization and classification of community's knowledge. In order to take knowledge as a resource, it is first necessary to understand the characteristics of community's knowledge and its different theoretical classifications and related types. The aim of this framework is to provide a definition of the vocabulary and the related semantics for the identification, categorization and classification of knowledge assets. Then these definitions will be useful to identify the characteristics of the community's shared knowledge (section 2.3.2)

First of all, I will make a clear distinction among the terms *data*, *information*, and *knowledge*, because a major problem with KM is the fact that despite intensive academic discussion on the terms data, information, and knowledge; in practice, they are often used in an uncoordinated way [Muller99] and they are even used synonymously many occasions. To provide a context, the following paragraphs present a definition of these terms and their relationships. These definitions have been taken from [Schreiber00]:

Data are uninterpreted signals that reach our sense every minute by zillions. A red, green and yellow light at an intersection is an example. Data represents specific properties of objects (entities and events in the real world). A set of object descriptions without a context remains data. Computers are full of data: signals consisting of strings of numbers, characters and other symbols that are blindly and mechanically handled in large quantities.

Information is data equipped with *meaning*. Data can become information when it is properly processed as structured data to serve a certain purpose. For a human car driver a red traffic light is not just a signal of some collared object, it is rather interpreted as an indication to stop. But the interpretation can be bit different depending on the custom of each country, for example in La Plata city a red traffic light means "stop" if there are cars crossing the street.

Knowledge is the whole body of data and information that people bring to bear to practical *use in action*, in order to carry out tasks and create new information. Knowledge adds two distinct aspects: first, a sense of *purpose*, since knowledge is the "intellectual machinery" used to achieve a goal; second, a *generative capability*, because one of the major functions of knowledge is to produce new information. It is not accidental, that knowledge is proclaimed to be a new "factor of production".

In short, *data* in context is *information*, and *information* that is applied is *knowledge*, i.e., experience transforms information into knowledge [Drucker98]. Mainly, in ks-communities it is important to have a clear distinction between information and knowledge. Information is a means of reducing uncertainty [Boersma96]. However, information can only reduce uncertainty if it adds something to knowledge, i.e., if what we learn with information contributes to improving both personal knowledge and the community's knowledge we form. Information is needed to act and make decisions but knowledge plays the pivotal role in making correct decisions and actions. According to Nonaka [Nonaka95] knowledge is a 'justified belief' (i.e. applied information) that increases the capacity for effective action.

Two general and related forms of knowledge classification can be found in much of the literature on KM. The first is the classification of knowledge into tacit or explicit; the second is the classification of knowledge into declarative, procedural or heuristic. The later introduces a third category to represent the location of knowledge in the organization: individual or group (collective) knowledge. Within these three categories, it is possible to fit forms of domain specific knowledge. The three categories are described briefly below [Vasconcelos01].

Tacit vs. Explicit Knowledge

Tacit knowledge is knowledge, which is experientially acquired. Polanyi [Polanyi96] used the phrase "we know more than we can tell" to describe what he meant by tacit knowledge. Tacit knowledge is a kind of knowledge that is very difficult to articulate with formal language because it is either too complex or simple because it is internalized in people's minds. Nonaka [Nonaka95] describes tacit knowledge as personal knowledge embedded individually in every human being, shared and exchanged in normal social interaction. It contains subjective insights and intuitions. The intellectual property of an individual, group or organization, and its culture are examples of tacit knowledge. Knowledge in the form of project experiences, task heuristics and human competencies that are difficult to capture and externalize are also examples of tacit knowledge within the work environment. In addition, there are two dimensions of tacit knowledge: the *technical dimension* and the *cognitive dimension*. Technical dimension encompasses the kind of informal and skills often captured in the term know-how. On the other hand, cognitive dimension consists of beliefs, perceptions, ideals, values, emotions and mental models; although they cannot be articulated very easily.

On the other hand, *explicit knowledge* refers to knowledge that is transmittable in formal and systematic language. It is the kind of knowledge that we can easily share and articulate because it is relatively independent of any particular individual or organizational group. Explicit knowledge can be articulated into formal language, including grammatical statements, mathematical expressions, specifications, manuals, etc. Explicit knowledge can be readily transmitted to others. Also, it can easily be processed by a computer, transmitted electronically, or stored in databases. Nonaka also conceptualizes the externalization of knowledge as a process of making tacit knowledge explicit (see section 2.2.2); before tacit knowledge can be communicated, it must be converted into words, models, or numbers that can be understood.

Nonaka & Takeuchi [Nonaka95] stated specific modes of knowledge creation within an organization using conceptual relationships between tacit and explicit knowledge, as it will be presented in 2.2.2, and this conceptualization of knowledge creation will be the theoretical framework for creating the conceptualization of the knowledge-sharing process and discussion process that is suggested in this thesis as the way of carrying out the activity of building knowledge (it will be dealt with in section 2.3.3).

Declarative, Procedural, and Heuristic Knowledge

Declarative knowledge is related to the physical aspect of knowledge. It is the knowledge type required in order to know what, who, where and when. It is essential in both interpreting and describing, from a certain viewpoint (conceptualization), the physical features of the world. It is knowledge of objects (entities or events) and consensual facts about the world, i.e. it is factual information about a given content area. For example, the knowledge represented in a data base conceptual schema or formal representations using first-order logic predicates or ontological representations of knowledge are examples of declarative knowledge.

Procedural knowledge is the knowledge required to accomplish a certain task: it provides a description of specific actions required to complete a particular task. It derives from the intellectual skill of knowing how to do something. Conventionally, procedural knowledge uses declarative knowledge to describe actions in step by step sequences. Procedural knowledge allows the representation of the behavior of a specific domain. An informal description of actions/steps, business rules, constraints, and exceptions that constitute a particular operation or organizational task is an example of procedural knowledge. A formal description representing that operation is an example of encoded procedural knowledge.

Finally, *heuristic knowledge* describes the knowledge related with work experience and implicit reasoning. As meaning depends on the individual's experience, heuristic knowledge grows with work experience. Heuristic knowledge is generated by an internal process and uses both declarative and procedural knowledge to solve problems and consequently to answer the question why [Davies03]. Heuristics gained during work experiences assist the resolution of future tasks. Heuristic knowledge can be interpreted as a specific type of tacit knowledge, as is hard to capture and externalize.

Individual Knowledge vs. Group Knowledge

According to Vanconcelos [Vasconcelos01], a third category of knowledge can be introduced to represent its location within an organization. Previously, it was stated that tacit knowledge is seen as a property of individuals. However, research has made it clear that a team of interacting individuals can have knowledge that transcends the knowledge of each of them individually [Walsh95]. Walsh uses the term *Knowledge Structure* to describe a "mental template" an individual imposes on an information environment to give it form and meaning. A mental template consists of organized knowledge about an information domain. This helps individuals to interpret and make sense of their environment and activities. Knowledge structures are built on past experience and are used to store data

to allow subsequent interpretation and action. Hence, individual knowledge is concerned with personal knowledge structures, while group knowledge is related to organizational knowledge structures. We can define *group knowledge* as the knowledge and skills acquired collectively by individuals who have been exposed to similar job-related situations [Reuber90]. For example, groups in a organization have part of their knowledge codified in workflow 'metaphors' that only the members of that group can understand. These workflow metaphors are typically the result of systematic communication practices that occur in the workgroup environment. According to Buckingham Shum [Buckingham97], group knowledge is multidisciplinary, hard to formalize, and generated "in discussions" with competing viewpoints. Therefore, any approach to support the collaborative development of a KR which represents the ks-community knowledge (group knowledge) should support the discussion occurrence.

2.2.2 Traditional Knowledge-Sharing Approach

A mode of knowledge-sharing, which has been largely discussed is *organizational learning*. Organizational learning is a process by means of which the knowledge held by the individuals is amplified, internalized, and externalized as part of an organization's knowledge base [Nonaka95]. According to Nonaka, organization learning is explained based on the knowledge creation spiral process. In his spiral, knowledge moves upward in an organization, starting at the individual level, moving up to the group level, and then up to the organization level. He also distinguished between explicit and tacit knowledge. Tacit knowledge becomes explicit knowledge by dynamic interactions among four modes of knowledge conversion [Nonaka95]. Organizational knowledge creation occurs when all four of these modes form a continuous spiral. The four types of knowledge conversion are:

- *Socialization -from tacit knowledge to tacit knowledge-* denotes the experiential processes by means of which tacit knowledge is shared among individuals. Communication between partners at meetings would be regarded as socialization.
- *Externalization -from tacit knowledge to explicit knowledge-* occurs when tacit knowledge takes an explicit form in written statements, models or metaphors. Externalization is highly imperfect and requires *reflection* among individuals to arrive iteratively at the correct representation. Through the use of metaphorical dialogues, individuals and groups "articulate their own perspectives, and thereby reveal hidden tacit knowledge that is otherwise hard to communicate" [Nonaka94].
- *Combination -from explicit knowledge to explicit knowledge-* ties together different bodies of explicit knowledge held by individuals through processes such as meetings, telephone conversations, and document exchanges. This reconfiguration of existing knowledge can lead to the creation of new knowledge [Nonaka94].
- *Internalization -from explicit knowledge to tacit knowledge-* explicit knowledge is converted to tacit knowledge. Internalization occurs through some form of 'learning by doing' activities in which concrete, articulated concepts emerge in iterative trial-and error processes.

This spiral of organizational knowledge creation allows the emergence of varied knowledge, through numerous interactions at different levels: (individuals, groups, organization). Although Nonaka's approach is oriented to organization learning in this thesis I will focus only on the individual and group learning, as it will be exposed in section 2.3, and in particular I will put emphasis on the fact of knowledge externalization in order to create a KR.

Nonaka and Konno, in more recent works [Nonaka98], claim that ks-activity is also strongly related to the surrounding environment. They introduce the concept of *ba* (a Japanese word that in English means *place*), to denote a shared space for emerging relationships. It can be a physical (an office, etc.), virtual (e-mail, teleconference, etc.), or mental (shared experiences, ideas, ideals) space or multiple combinations of these spaces. *Ba* is considered as shared space which is used as the foundation for the creation of (individual and/or collective) knowledge.

These *places* could be identified in numerous organizations: a project team, a specialized team, a department in a firm, etc. In many cases, a group of individuals has various skills, that is, know-how and/or knowledge. Nonaka and Konno believe that the exchanges between the different members confer an increase in knowledge and that the *ba* is therefore a kind of intermediary state that is in constant evolution and which, according to the situation, has a temporary or permanent character. In consequence, the *ba* can be the place where to create a form of group memory. The *ba* can generate formalized elements (knowledge base, journals, etc.), forms of memory (KR) in their own right. However Nonaka's approach does not provide any clue about what the better approach to develop this KR is like.

2.2.3 KR Developing: Distributed Approach

Strongly related to the act of sharing knowledge it is the development of the KR. The KR puts up the externalized knowledge. Traditionally, KM field has been in charge of the development of KR system [Abecker99] or also called knowledge repositories for capturing and sharing knowledge. There are many approaches to develop KR, most of them follow the traditional KM approach, where a KR system is a knowledge repository that stores the group knowledge and is administrated by knowledge managers. The knowledge managers' tasks are mainly to capture community's knowledge and store it. In this approach, KM considered organizational cognition as *a convergent process that collects peripheral "raw" knowledge, from various sources, and codifies it into a central repository* [Bonifacio02b]. As a consequence, technology is viewed as a tool for enabling central control, standardization, high capacity, and robustness. In this *centralized* paradigm, the organization managers, supported by knowledge engineers, collect and structure the contents of the organizational memory that will then spread, expecting community members to use it and update it. Most KM projects aim at creating large, homogeneous knowledge repositories, in which corporate knowledge is made explicit, collected, represented and organized, according to a single - hypothetically shared - conceptual schema. Such a schema is meant to represent a shared conceptualization of corporate knowledge, and thus to enable communication and knowledge sharing across the entire organization.

On the other hand, *distributed* approach emerges as an alternative to avoid the disad-

vantages of the centralized approach. KM community has recently stopped worrying about knowledge organization and starts worrying about collaboration among people. This community is becoming the focus of KM in the management and nurturing of collaboration between peoples [Dignum03]. But, due to community's nature, communities develop their own memories. It is a shared responsibility that involves a collaborative process, which aims at building the shared knowledge. There are many community-oriented technologies for supporting community's knowledge sharing as they are showed in [Wenger01], in general they are groupware-oriented. One of the main challenges of these tools is to reproduce the dynamism that the community has around the knowledge. The community's knowledge is constantly growing and evolving, due to each community memory contribution stimulates new knowledge emergence and therefore keeps the community "in action" sharing knowledge.

In [Bonifacio02a], Bonifaccio stated that

"the concept of *absolute knowledge*, which refers to an ideal, objective picture of the world, leaves the place to the concept of *local knowledge*, which refers to different, partial, approximate, perspectival interpretations of the world, generated by individuals and within groups of individuals (e.g. organizational units, groups) through a process of *meaning negotiation*, namely a process of *distilling* a schema which makes sense for that unit".

Therefore, *local knowledge* is a matter that is continuously negotiated by people that has an interest in developing a common perspective, and in understanding the different perspectives of the world that exist in the group. Because of that, knowledge is seen as a *heterogeneous and dynamic system of multiple "local knowledge systems" that live in the interplay between the need of sharing a perspective within an organizational unit (to incrementally improve performance) and of meeting different perspectives (to sustain innovation)* [Bonifacio02a].

The *distributed* paradigm represents organizational cognition as a *distributed process that balances the autonomous KM of individual and groups, and the coordination needed in order to exchange knowledge across different autonomous entities; from this perspective, technology is viewed as a way of enabling distributed control, differentiation, customization, and redundancy* [Benerecetti00]. The distributed-KM approach [Bonifacio02b] is based on two principles:

1. *Principle of Autonomy*: each unit should be granted a high degree of autonomy to manage its local knowledge. Autonomy can be arisen at different levels. Mainly it points to semantic autonomy, this means the possibility of choosing the most appropriate conceptualizations of what is locally known (for example, creating its own knowledge context);
2. *Principle of Coordination*: each unit must be enabled to exchange knowledge with other units not by imposing the adoption of a single, common interpretative schema (this would be a violation of the first principle), but through a mechanism some mechanism that allows managing and negotiating the different perspectives.

In the context of distributed-KM approach, technology plays the following roles:

- giving to each unit the possibility of representing and organizing knowledge according to its goals and interpretative perspective;
- providing tools to support the exchange of knowledge across different units without assuming shared meanings, but rather enabling the dynamic exchange of different meanings;
- setting mechanisms and protocols to enable, through cooperation, the emergent and bottom-up formation of the shared conceptualization
- as complex organization are seen as a constellation of units (individual or groups), the "socially distributed architecture" has to be design as an "architecturally distributed" computer-based system for supporting KM processes.

2.3 The Knowledge-Sharing Activity

The ks-activity is a collaborative learning process by means of which the community accumulates knowledge [Diaz04a]. This process is carried out in an iterative and incremental way as the community knowledge is being built [Stahl05a].

While a community is sharing knowledge, its knowledge is constantly growing and in evolving. Knowledge evolves as a consequence of participants' contributions. Community's members are the participants of knowledge evolution, because each new knowledge contribution to the community is a step forward to a new state in the knowledge building.

Knowledge contribution is the fact of communicating publicly any knowledge to the community. This knowledge can be of many different nature (see the community knowledge section 2.3.2). Besides, a knowledge contribution means bringing knowledge from *individual knowledge context* to the *community (or shared) knowledge context*.

The *knowledge context* represents the knowledge cognition of an individual or a group [Stahl05a]. The community knowledge context is also known as the common understanding of a group of people. In particular, I differentiate between the community knowledge context that represents the common understanding, and the individual knowledge context that is the personal knowledge. This differentiation is because both cannot coincide, they can be "divergent", its responds to the concept of local knowledge context defined by Bonifacio in [Bonifacio02a] which has been mentioned in section 2.2. When individuals share knowledge, they are constantly going from/to individual knowledge context to community knowledge context.

The ks-activity is a collaborative learning process through out the community develop its own common understanding –*shared knowledge context*. KS-activity can be seen as a spiral process where knowledge goes emerging in each cycle (see KS-Process section 2.3.3). This process describes an augmentative building of a common understanding through the contribution of "knowledge". People always add more knowledge in each contribution, whatever this contribution means.

However, this process is not a lineal process, because it also involves the discussion of the different positions. Discussion is possible when contributions do not only mean

to contribute with "knowledge" but they also mean sharing different perspectives of the same subject. Therefore, the KS-process has to cover the knowledge building activity as a discussion activity where meaning negotiation takes place. This meaning negotiation allows being distilled the common understanding of the community.

This approach states the ks-activity as a design activity where the community knowledge context is developed. Therefore, as any design activity, it states a "wicked problem" [Rittel73]. It is well-known that the fundamental way in which wicked problems are tackled is by discussing them. Consensus emerges through the process of laying out alternative understandings of the problem, competing interests, priorities and constraints. The application of more formal modelling and analysis tools is impossible before the problem can be articulated in a concise, agreed upon, well-bounded manner. Therefore, the knowledge discussion activity is a relevant part of ks-activity (see 2.4). Knowledge discussion activity is in charge of facilitating the meaning negotiation by allowing the occurrence and coexistence of divergent knowledge positions at the shared knowledge context. Argumentation-based mechanisms are frequently used to allow the discussion development.

Any technological approach for supporting ks-activity should then offer suitable mechanisms to support it. To achieve this goal, it is important to have a complete understanding of the kind of knowledge the community shares. Besides, significant attention must be paid to the process that allows communities to share what they know in a coherent way across their activity. In the following, first, a scenario is presented as an example, where knowledge sharing occurs in a knowledge-sharing community and it is described what kind of knowledge the community shares. Then, section 2.3.3 introduces the knowledge sharing process without taking into account the knowledge divergence occurrence. It is delayed to the section 2.4, where the knowledge divergence is presented as a cognitive conflict at the common understanding. Even in this section, divergence occurrence is understood as the posting of alternative position in the context of the knowledge sharing process. Finally, as consequence of this analysis, it is possible to find out the specific CSCW requirements of a distributed KR system that supports the ks-activity; they are exposed in section 2.5.

2.3.1 A KS-Activity Scenario

This scenario is based on a common knowledge-sharing situation that usually takes place at the Groupfia community. Groupfia brings together people interested in learning, using and building groupware tools. Groupfia's members are mainly people that work or have worked at the Lifa Lab ¹, and may be co-located or not:

Ale, which up to this moment is working in Germany, has just found out *tiki-wiki*², a *wiki* environment with a forum, e-mail, etc. He has sent an e-mail to the community announcing his discovery; consequently from Argentina, Fede sent back to the community list a message where he explains he already knows this environment, and even his group is already using it. And Miguel, who

¹Lifa, Computer Science Research Laboratory at La Plata University. www-lifa.info.unlp.edu.ar

²<http://tikiwiki.org/>

belongs to the same group as Fede, has shared a URL with tikiwiki information. Rick has also said that *tikiwiki* is equivalent to *JSPwiki*³ (another groupware tool) since it has comparable functionalities to *JSPwiki*; whereas Diego, from France, has said that tikiwiki is not exactly equivalent to JSPwiki, because although they share many functionalities, they do not share all of them. Meanwhile, Guille asks where he can find similar groupware tools that the community knows or has already evaluated; consequently, Rick has answered that this question can be forwarded to Fede, because he is definitely an expert in wiki tools.

For non-expert readers words like *wiki*, *forum*, *JSPwiki* and others may have no meaning, however, for Groupfia members, they are full of meaning since they are part of the daily Groupfia's vocabulary or common code. This is because these terms are part of the community knowledge context. In the following sections I will refer to this scenario to introduce ks-activity features.

2.3.2 The Community Knowledge

KS-communities are knowledge-intensive because all of their activities are the knowledge building; the most frequent activities consist of knowledge exchange among community members. As a consequence of this activity, the community accumulate knowledge. Accumulated knowledge is the knowledge that the community already possess, and hereinafter the *community knowledge*. The community knowledge corresponds to the community shared interest and it also represents the common understanding on this domain of interest.

In terms of the classification of knowledge given in the section 2.2 of this chapter, community knowledge is group knowledge and most of it is declarative. However, its nature is varied. Communities do not only share knowledge about a topics of interest, they also share knowledge about who are participating in the community, who knows what, who is interested in what, level of expertise, perspectives, among others. All of this knowledge is integrated in the community knowledge. Following, we classify and detail community knowledge according to their nature:

Domain Knowledge. This is knowledge about the domain of interest of the community and it consists of conceptual elements and facts that conceptualize the domain of interest and competence. Community domain knowledge also represents a consensual knowledge and shared common language. In this particular case, domain knowledge is not only declarative, but also it may be procedural and even heuristic. Next, some examples of domain knowledge were extracted from the scenario:

- tikiwiki, a wiki environment with a forum, email, etc
- tikiwiki is equivalent to JSPwiki

Social knowledge. This is knowledge about members and their organization. Member's knowledge is knowledge about who is each member. Members can be individuals

³<http://www.jspwiki.org>

or groups. In a community, people gather moved by particular interests; they get together in smaller communities. Each member can participate in different groups. Groups can be established by some grouping criteria; groups mainly make up due to common knowledge interests, but they can also follow social interests like affinity, confidence or others. These interests are causes for shaping groups. For example:

- Fede, Miguel, Ale, Diego, Rick, Guille are well know as community member
- There is a group to which Fede belongs
- Miguel who belongs to the same group as Fede

These two sources of knowledge are the first knowledge that can be formalized. However there is another source of knowledge, it is knowledge about the relationships that are established between the domain knowledge and community members. A community member is related to a piece of knowledge by different causes; for example because it has a certain level of competence about a particular subject, or it has a responsibility to respect a subject (plays a role), is interested in a subject, or has a particular perspective or opinion about the subject. While a community shares knowledge, new relationships between knowledge artefact and people are established, for example relationships of interest (a member is interested in this knowledge); relationships of expertise (a member is expert on this concept); relationships of ownership (this concept is a contribution of this member); and relationships of privacy (this concept is private to this member). These relationships are nurtured during the community activity and most of the time this knowledge is an implicit knowledge that is hard to externalize, but it, in fact, exists. Therefore, it is defined a third category of community knowledge:

Members Profile. This is knowledge about the relationships that exist between the knowledge and people. This knowledge describes the interest, capabilities, and expertise of the community members. More complex relationships are also held between domain knowledge and community members, i.e. knowledge of the knowledge that members have i.e. "who knows who knows what", for instance. Next expressions show examples of member profiles knowledge:

- Ale is interested in tikiwiki: this is held because he contributes with this knowledge
- Fede knows tikiwiki: this is held by two assertion because this is said by himself and because Rick also says this
- Rick say more, he says that Fede knows everything about groupware tools.

Community knowledge can be seen as conceptual network made up of conceptual knowledge artefact (domain and members knowledge) linked by associations among them. These association can answer to different causes: intrinsic conceptual relationships between domain knowledge artefact, or special relationship between domain knowledge artefact and people, people and people, and more.

2.3.3 Knowledge-Sharing Process

The knowledge-sharing scenario presented above shows how the community carries out an iterative and incremental knowledge-sharing process. This process begins when Ale found out the existence of "tikiwiki, a new kind of wiki" and decides to share it with the community; and therefore he "write an email and sends it to the community mailing list". Then, the process continues when other members realize this, and begin to send comments, and additional information that allows the community to have a more complete idea of what a "tikiwiki" is (see Fede, Diego and Rick comments).

However, when a distributed KR system is used as computer-support for the ks-activity, it is necessary to understand the process that a community carries out to build this KR collaboratively. This process has to show how the knowledge is exchanged among the participants and how knowledge is converted into tacit or explicit knowledge. When the ks-activity is computer-supported by the collaboratively development of the KR, it is possible to remark that:

- knowledge moves from private knowledge contexts to the community knowledge context and comes back to individuals again. This is an intrinsic characteristic of the ks-activity, but it gains relevance when the distributed approach is used to develop the KR.
- knowledge is no longer tacit to become explicit and then become tacit again. It is a characteristic derived from the decision of simulating the ks-activity by a distributed KR development.

In this thesis I suggest the knowledge-sharing process as the process to describe how a knowledge-sharing community shares knowledge at the same time that it develops its own KR. I called this process, the *knowledge-sharing process* (ks-process) and it is a spiral process, made up of 4 steps: externalization, publication, internalization and reaction.

Externalization - from tacit to explicit knowledge and from private to private knowledge context

Externalization is an individual and private activity through which an individual makes explicit some tacit knowledge that is at her/his individual knowledge context. In fact, when Ale wrote: "tikiwiki is a wiki tool" he has externalized his own knowledge.

Some knowledge representation system is needed to make the knowledge explicit. This knowledge representation can be informal or formal, going to informal systems like those supporting socialization (emails or document writing) to semi-formal systems like OntoShare [Davies03] that classifies documents by an ontology; or even to only formal systems (i.e. to develop a knowledge base with ontologies). Specially, I am interested in those approaches that allow obtaining a formal representation of the shared knowledge. The knowledge representation system allows identifying *knowledge artefact*. A knowledge artifact is a unit of knowledge represented by the knowledge representation system.

Submission/Publication - from explicit to explicit knowledge and from private to shared knowledge context

Publication is the act of making public a new knowledge at the community level. Submission corresponds to the transfer of some knowledge from the individual to the community knowledge context. Publication has externalization as pre-condition. The act of submitting an externalized knowledge (knowledge artifact) is generally meaning as a *contribution* and the subject of the contribution is the attached knowledge. When, Ale sends an e-mail to the community with his externalization, he has submitted his own contribution.

Internalization - from explicit to tacit knowledge and from shared to private knowledge context

Internalization is an individual and private process, which takes place when individuals realize and acquire the subject of a new contribution - individual learning. In this process, internalization is considered similarly Nonaka [Nonaka95]. Internalization makes possible that knowledge goes from community knowledge context to part of the individual knowledge context. Internalization is not easy to detect, but it is true that if there is reaction, then there has been internalization. There were many examples of internalization in the example. We can affirm that Fede, Diego and Guille internalize the new knowledge subject because they have replied (reacted) to Ale contribution by performing some action.

The ks-process is an incremental and cyclical process. It is incremental because each publication makes the KR grow. And, it is cyclical because these steps are repeated constantly. Therefore, the execution of this process implies that the building of knowledge describes a spiral process where knowledge emerges in each cycle.

However, the ks-process sometimes describes a cycle, because an internalization occurrence motivates a new externalization and publication. This is the case when someone reacts to a previous contribution. To describe this particular situation, I also consider reaction as part of the ks-process.

Reaction is the act of giving some kind of response to a previous contribution. Any reaction is an externalization of an individual position before a new contribution. Reactions are always tied to an initial contribution and always give an "augmented" version of the original knowledge subject because it is improved with new knowledge. Reactions are interesting to observe because they imply that internalization has previously taken place and it "closes" the spiral cycle of the ks-process.

Reaction can be *private*, this means that it only produces some change at individual knowledge context; or it can be *public* when it affects community knowledge context. Both reaction kinds imply an externalization, but only public reaction involves a publication. Fede, Diego and Guille have reacted in an active fashion.

The KR is developed through the conversion of knowledge that proposes the ks-process. This conversion is a cycle and occurs in two senses: tacit-explicit and private-shared.

- Tacit-private knowledge becomes explicit-private through externalization.

- Explicit-private knowledge becomes shared-explicit through publication.
- Shared-explicit knowledge becomes tacit-private through internalization.

As reaction can be understood as an externalization/publication, it can be also comprised in the knowledge conversion cycle. However, any conversion by reaction is triggered from a initial (previous contribution) and the ks-process's cycle describes the discussion thread (see section 2.4.2).

Another characteristic of this ks-process is that in each cycle it produces a new augmented knowledge version; because publication is augmentative, each publication brings more knowledge to the shared knowledge context. Therefore, while the community is sharing knowledge, its knowledge context is constantly growing and in evolution. Each new contribution to the community knowledge is a step forward to a new community knowledge state. Knowledge evolution occurs in long term but as a consequence of a ks-process execution.

The ks-process guarantees the reflection among individuals which is mandatory to achieve a common understanding. Figure 2.1 is a scheme of the ks-activity carried out in the previous scenario. Bubbles represent the different knowledge contexts and each one is identified by its owner, and in particular, the central bubble represents the shared knowledge context. This ks-activity is triggered by Ale's externalization (sheet shape) and it follows by the publication (think black arrow). Next, the internalization takes place and it is represented by dashed arrows. Lastly, reactions are shown as grey arrows. One can notice that it is not possible to distinguish, in this scheme, that Diego's contribution is a reaction to Rick's contribution.

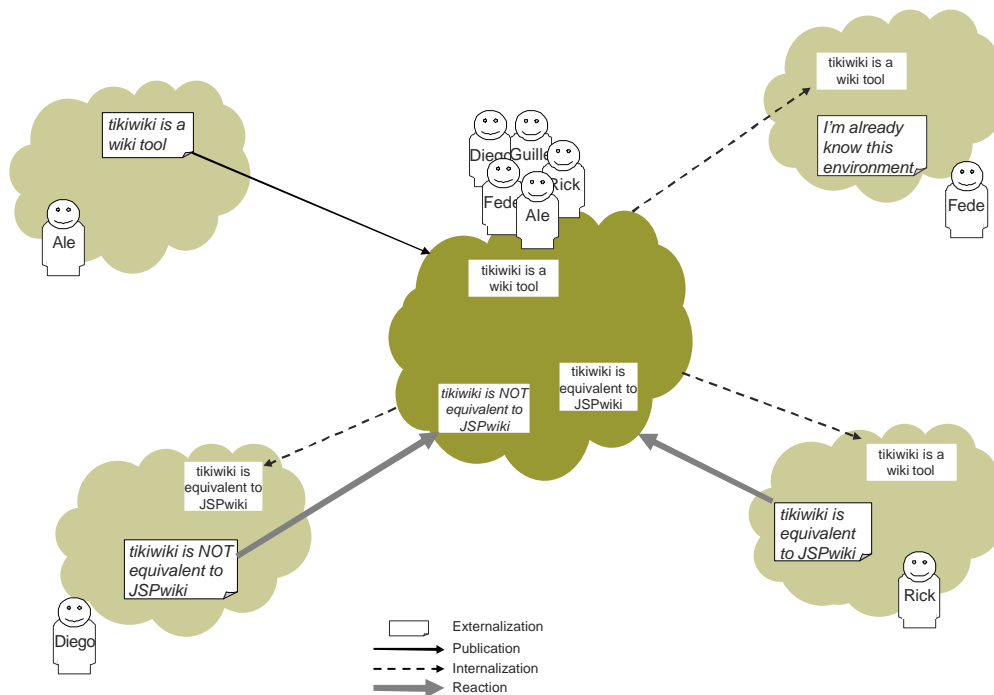


Figure 2.1: A schematic representation of the ks-activity carried out at scenario

2.4 Knowledge Divergence Occurrence

Both, Nonaka's knowledge sharing approach and distributed KR developing paradigm manifest the importance of allowing community members to be able to express different knowledge perspectives. Although, finally ks-communities accumulate knowledge and develop a unique perspective on their topic of interest, before reaching this unique perspective, divergent knowledge positions appear as a natural consequence of the act of sharing knowledge. In ks-communities, it is not so realistic to think that everybody agrees with everything that is told; whereas it is very often to observe people that express different positions or argumentations in the context of the same knowledge subject. These different perspectives or points of view about a topic of interest reveal knowledge divergences at the community knowledge context. To *coexist* with knowledge divergence is very natural in any knowledge-sharing community Knowledge divergence is generally considered as a *conflict* at the common understanding that reveals a cognitive conflict. Despite the fact that cognitive conflict occurrence can be seen as unfavorable situation in knowledge-sharing community, I claim that it is possible to take advantage of this situation, because it exactly describes the discussion activity in which the participants are involved to learn collaboratively [Stahl05a]. This discussion activity describes an evolutionary process as the knowledge is built (emerges). In this sense, the knowledge building represents the developing of a common understanding by means of the meaning negotiation.

Knowledge divergence arises because of differences among individuals. Individuals' experiences, personalities and commitment become the potential for cognitive conflicts.

2.4.1 Cognitive Conflict

The word conflict has been used for many propose. Easterbrook citing Flink in [Easterbrook93] remarks the existence of different uses of the term *conflict* in the literature reflecting the so many different conceptual frameworks to study "conflict". There is a variety of terms that are used to refer to conflict, i. e. conflict, competition, tensions, disputes, opposition, antagonism, quarrel, disagreement, controversy, violence, conflict resolution, mode of resolution; all of them refereing to different aspect of a conflict. However, Putnam and Poole's definition [Putnam87] has been recently adopted because it is a broad definition and subsumes a variety of situations. According to Putnam and Poole,

a *conflict* is the interaction of interdependent people who perceive opposition of goals, aims, and values, and who see the other party as potentially interfering with the realization of these goals.

This definition mainly emphasized three characteristics of conflict: interaction, interdependence and incompatible goals. From a CSCW point of view, we can claim that this definition does not oppose conflict to cooperation, but it states that conflict may arise whether people are cooperating. Besides in the context of ks-communities this definition also applies when "goals" are understood as "meaning" because these communities are also characterized by: interaction, interdependence and incompatible meaning. Incompatible meaning refers to knowledge divergences or different perspectives and point of view that are exactly incompatible positions. This is what I mean by *cognitive conflict*.

In [Easterbrook93], Easterbrook says: "the more interaction there are between people, the more opportunities there are for conflicts". In addition, it is possible to say that *the more ks-activity there are between people, the more opportunities there are for cognitive conflicts*.

When conflict has to be managed there are some features to be taken into account to understand suitably the conflict characteristics. Conflict characteristics are related to the conflict process, structure and tasks.

- The *conflict process* focuses upon the sequence of events within a conflict episode, and it is intended to be useful when intervening directly in the stream of events of an on-going episode. For example the *Reaction process models* ([Patchen70]) describes each action in terms of a reaction to the last action of the other party, according to various characteristics of each party. Reaction at the KS-process exactly is useful to describe this kind of conflict process.
- The *conflict structure* focuses upon the conditions which shape conflict behavior in a relationship, and is intended to help to restructure a situation to facilitate various behavior patterns. The discussion thread model that I will introduce later captures the structure of the cognitive conflict based on an argumentation pattern.
- The types of tasks that small groups carry out are also considered to characterize conflicts. Among all types of tasks, McGrath, in [McGrath84], identifies that tasks around "*conflicts of viewpoint*" ('cognitive conflict') can be classified in the *negotiating* category. In negotiation task conflict is prevalent. This is the reason which motivates me to manage group cognitive conflict as a negotiation activity. Although in this research the negotiation task is more dedicated to allow the conflict development, than to resolve it.

When the knowledge-sharing community is computer-supported, there is a set of assertions about conflicts grouped according to the aspect of conflict that they refer to. Cognitive conflicts are characterized by the following assertions:

- the factors that affect whether conflict will arise (occurrence); because divergence occurrence is inevitable. The more ks-activity there is between people, the more opportunities there are for cognitive conflict (divergence occurrence).
- the specific causes of conflict (causes); as it was explained above, divergence occurrence is inherent to the ks-activity. Besides, because technological mediation introduces conflicts, anonymity and physical separation contribute to conflicts.
- the role that conflict may play in group interactions (utility); because conflict can be productive. Divergence occurrence stimulates ks-activity, then it can be seen as a source for the emerging new knowledge
- the processes involved in an individual conflict episode (development), because conflicts are part of the ks-activity. Conflicts are developed/resolved by community participants, at least they make conflicts evolve.

- approaches to handle conflicts, including resolution techniques (management). Identification of opened conflicts and conflicts closing policies are requirements to the conflict management.
- outcomes and long term effects of conflict (results).

These assertions allow designers to count with a framework to design suitable software solutions to support the collaborative activity. In particular when this collaborative activity is ks-activity, they help to situate the problem of cognitive conflicts.

Although the achievement of a consensus may or not happen, what actually matters is the process that takes place while the community persists with a conflict. Consensus is achieved through the conflict evolution. In consequence, to support divergence occurrence (conflicts at the common understanding) is fundamental to support suitable ks-activity. In this thesis, I do not focus on how community reaches the consensus, but how it coexists with the conflict while it shares knowledge, because I believe that suitable support of the coexistence with the conflict is a good way to allow the community to solve the conflict.

Knowledge divergence can occur privately or publicly. As it was defined above knowledge divergence means a conflict at the common understanding. However, this definition does not say anything about the way in which conflict is stated in the context of the community. Conflict occurrence appears firstly as a private activity and then it may become a public activity. A community member can be in conflict with the common understanding because he/she has divergent points of view or perspectives regarding it. The conflict can remain at private knowledge context, while user does not communicate it to the community. Conflict publication makes conflict arise in public knowledge context.

In this thesis I follow a procedural model of conflict, where it focuses on the sequence of events within a divergence occurrence. In particular, I focus on a reaction process model, which meets the ks-process described in section 2.4.2, where some knowledge contributions are understood as reactions to a previous contribution of another party.

2.4.2 Reaction as a means for revealing discussion activity

In the context of the ks-process, knowledge contributions can be spontaneous or they can be caused by a reaction. A spontaneous contribution is a contribution that was not motivated by any other previous contribution. On the other hand, a contribution by reaction is a contribution motivated or stimulated by a previous contribution of another party. A contribution by reaction is more than an externalization followed by a publication, it reveals a discussion activity. Contributions by reactions describe a "chain" (not necessary a lineal chain) of contributions which reflects the *reflection* among members in order to iteratively reach a consensus of the common understanding. Therefore, reaction is the source for revealing the discussion activity.

Knowledge contributions can be classified according to their role in the discussion as initial contributions or contributions by reaction. An initial contribution is a spontaneous contribution that becomes initial when a reaction to it occurs. A contribution by reaction is a contribution that is caused as a reaction to the initial contribution.

A contribution by reaction really is important by its motivation. There are many causes why a reaction occurs. It can be motivated to complement a previous contribution, or to give another point of view or just to provide arguments for the original contribution. Those reactions that give other points of view introduce divergence, stating a cognitive conflict at the common understanding. Therefore, the occurrences of knowledge divergence are manifested as reaction, and this is the importance of the reaction step in the ks-process. Reaction step is responsible for allowing discussion activity which is covered by the ks-process.

A contribution by reaction is always attached as successor of another contribution, either initial or by reaction. Initial contributions represent the starting point of a discussion thread. Any contribution is a potential initial contribution; they become as such just when a contribution by reaction appears attached to it. The initial contribution states the subject of discussion.

In this analysis I cannot forget the fact that reaction can be passive, this means that there is not a real contribution, because it only involves externalization without submission. But I cannot say that there is no divergence occurrence. It is possible, in order to both respect member individualism and privacy (sometimes members only express divergence as a private act) and consider it as temporal state that in the future will be or not submitted.

In order to establish the knowledge discussion activity as a complementary part of the ks-activity I suggest to model the discussion thread. Figure 2.2 partially shows the discussion thread of the scenario of section 2.3; which manifests the linkage among the different contributions. Rectangle shapes distinguish different kinds of contributions; where rectangles represent contributions, but, in the context of the discussion thread, an initial contribution is a bold rectangle, an augmentative one is a regular rectangle, a divergence is multi-rectangle and an argumentation is a dashed rectangle. In this example it is shown that Diego's contribution is a conflictive contribution to the Rick's contribution. For example, when Rick says *tikiwiki is equivalent to JSPwiki*, he is giving an augmentative version of the tikiwiki concept; this contribution complements Ale's previous contribution: *tikiwiki is a new wiki*. However, when Diego says that *tikiwiki is not equivalent to JSPwiki*, he contributes with a divergent contribution to Rick's contribution. All of them are always attached to some previous contribution. There are also an argumentation example, it is when Diego states that tikiwiki it is not equivalent to JSPwiki because although they share much functionality, they do not share all of them.

2.5 CSCW Approach to Support KS-Activity

There are many computer-supported approaches to ks-activity. Wenger enumerated in [Wenger01] many technologies already used by on-line communities like home pages, on-line discussion groups, collaborative-shared workspaces, document repositories. There are also many technologies for supporting ks-activity where the community develops its own KR. However, there are not many systems explicitly oriented to knowledge discussion and knowledge divergence occurrences. The existing ones only focus on one or more aspects of the whole picture.

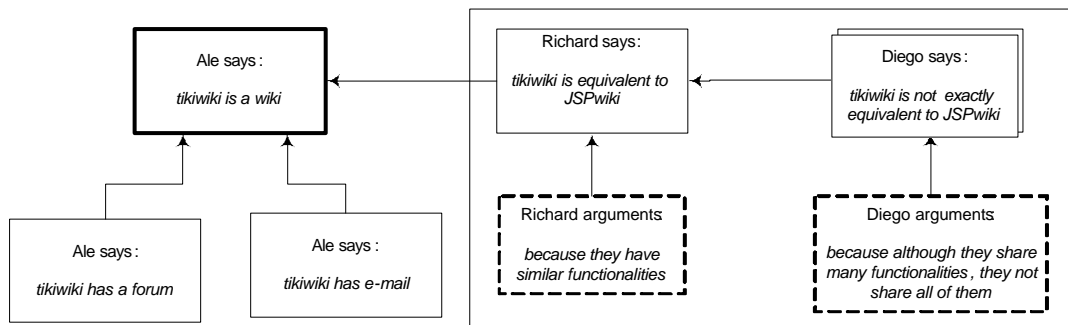


Figure 2.2: The discussion thread of the scenario

In this thesis, I suggest, as a computer-support for the ks-communities, a groupware system that is a distributed shared KR system that support the ks-activity, paying special attention to the discussion activity and the knowledge divergence occurrence. This system is characterized by

- *the collaborative developing of the KR*, each participant can contribute with knowledge. From the community's point of view, community knowledge is the shared knowledge that the community develops collaboratively.
- *preserving the autonomy principle*; I differentiate between the *community knowledge context* that represents the common understanding, and the *individual knowledge context* that is the personal knowledge, because both cannot coincide. When individuals work in a community, they are constantly going from/to individual knowledge context to community knowledge context.
- *occurrence and coexistence of knowledge divergence*, each member can express individual points of view, even if they cause cognitive conflicts.
- *monotonic extension of the KR*. Each participant contribution must be augmentative; a contribution always adds more knowledge, even if it is a divergent contribution. Nobody can cancel knowledge that was already shared. Augmented contribution corresponds to the concept of monotonic extensions of Michael Klein [Klein01].

2.5.1 CSCW Requirements to Support KS-Activity

In particular, I am interested in those that allow the community to develop its own community's repository with the capability of expressing divergences. The community repository is a CKS system where the the community collaboratively develop a conceptualization of this domain of interest by sharing knowledge and expressing divergences. For achieving this, the community needs a knowledge sharing workspace that supports the ks-process with the following requirements:

- *Representation of individual and community knowledge context*. People need to differentiate between private and shared knowledge. The individual autonomy is

preserved by the manipulation of two integrated workspaces. One to represent the individual knowledge context and the other to represent the public workspace. Each workspace support different knowledge sharing actions, while knowledge externalization is a private action, publication is a public one.

- *Knowledge Divergence Management.* The occurrence of conflict has to be considered as part of the computer-supported collaborative activity, in opposition to traditional approaches where conflicts are managed outside of the system. In order to support the dialogue among the community members; it is necessary a contribution-classification approach to model knowledge discussion activity with the occurrence of divergence. The discussion thread is the mechanism for expressing conflicts explicitly as a monotonic extension of the KR. This model have to capture the nature of the contributions by the reaction that takes place in the ks-activity that allows tackling the collaborative development of the KR.
- *Knowledge representation formalism.* For externalization, it is mandatory to define a mechanism that allows one to make the knowledge explicit. This formalism is embedded in the knowledge sharing workspace and define the type of the allowed actions. This requirement is not a CSCW requirement but ks-activity and the discussion activity will be dependent on the knowledge representation formalism. In this thesis I have selected ontologies as the knowledge representation system.
- *Knowledge and Discussion Awareness.* Internalization facilities are needed to have a suitable awareness of knowledge and discussion evolution. One of the main challenges of a technological support for ks-communities is to stimulate a dynamic activity around the knowledge. This means maintaining a high level of interaction among the community members. It is well studied in the CSCW literature that *awareness* is the resource to keep the group engaged into the collaborative activity, that is the awareness which helps to increase the level of interactions. Then, awareness can be understood as a suitable means to share knowledge; it would work as a vehicle to push this activity. Although a priori "more interaction implies more conflicts" is not a reflexive relationship, suitable awareness can make this expression a reflexive relationship, and consequently achieve more knowledge emerges.

2.6 Conclusion

Based on the previous requirements in the following chapters I will propose a knowledge sharing framework which models a groupware application that allow ks-communities to share knowledge. This framework has to conceptualize a knowledge-sharing workspace (ks-workspace) which supports the necessary functionalities to cover the ks-activities expressed in terms of the KR and the ks-process —externalization, publication internalization and reaction activities. This ks-framework has to support the collaborative development of the KR, but also it has to group in three different components the necessary functionalities. Components are thought in order to cover ks-process steps and provides the specific services to support the knowledge conversions discussed in section 2.3.3.

The ks-workspace component includes those elemental framework services to support the knowledge conversion features related with externalization and publication. Divergent management component, that is part of the workspace component, points to support reaction occurrences and pay special attention to the discussion thread where knowledge divergence takes place. The last component covers the awareness services that facilitate internalization. Particularly, there are two awareness services: knowledge awareness and discussion awareness. These components will be *knowledge-sharing workspace*, *divergent management component* and *awareness component* (a detailed description will be provided in Chapter 5).

Chapter 3

The Knowledge-Sharing Framework

Contents

3.1 Collaborative Knowledge-Sharing Framework	47
3.2 Knowledge Artifact	49
3.3 The Knowledge-Sharing Workspace	50
3.3.1 The Management of Knowledge Repository Versions	51
3.3.2 Private and Public Workspaces	53
3.4 Augmentative Development of the Knowledge Repository . .	55
3.5 Divergent Knowledge Management	58
3.5.1 Discussion Thread Model	59
3.6 Conclusion	63

This chapter introduces the knowledge-sharing framework, a CSCW approach to support ks-activity with divergence occurrences. This framework describes the fundamental components of a CKS system. These components are: the knowledge-sharing workspace where the KR is built, the divergence management component that allows the occurrence of cognitive conflicts, and the awareness component that keep people aware of the ks-activity. This chapter will be mainly in charge of dealing with the ks-workspace and the divergence management component. While the awareness component will be presented in Chapter 5.

3.1 Collaborative Knowledge-Sharing Framework

The collaborative knowledge sharing framework is a conceptual framework that joins fundamental components to perform the ks-activity in terms of the ks-process described in

section 2.3.3. These components are understood in the context of a groupware application through which a KR plays the role of the shared object; this means the ks-activity is reduced to the developing of the KR in a collaborative fashion by performing the ks-process, and thus, while the community shares knowledge, the KR is developed collaboratively [Diaz04b].

This framework is conceived as a shared workspace (the knowledge-sharing workspace) which supports the necessary functionalities to cover the ks-activities expressed in terms of the KR and the ks-process —externalization, publication internalization and reaction activities.

Macroscopically, the ks-workspace supports the collaborative development of the KR, but microscopically it groups the functionalities in three different components. These components are thought in order

- to cover ks-process steps and provide the specific services to support the knowledge conversions which were discussed in section 2.3.3:
 - Tacit-private knowledge becomes explicit-private through externalization.
 - Explicit-private knowledge becomes shared-explicit through publication.
 - Shared-explicit knowledge becomes tacit-private through internalization,
- to support the occurrence of cognitive conflicts. Knowledge divergences must take place at the ks-workspace level, in order to allow the discussion activity to become part of the collaborative activity.
- to keep people aware of the ks- activity which takes place in the ks-workspace, in order to make divergence evident.

These components are: the *knowledge-sharing workspace* (ks-workspace), the *divergence management component* and the *awareness component*. Knowledge-sharing workspace component includes those elemental framework services to support the knowledge conversion features related with externalization and publication. Divergent management component, that is part of the ks-workspace, points to support reaction occurrences and pay special attention to the discussion thread where knowledge divergence takes place. The last component covers the awareness services that facilitate internalization. Particularly, there are two awareness services: knowledge awareness and discussion awareness.

Knowledge-sharing workspace is a shared workspace where KR is built through the externalization and publication of the knowledge. The knowledge has to be represented following some criteria. By means of the knowledge representation system, knowledge is externalized. By publishing externalized knowledge people build the shared KR. As externalization is a private activity, the workspace differentiate between a private workspace, where each individual externalizes her/his own knowledge, and the shared workspace that collect published knowledge. This approach also guarantees that publications provoke an augmentative version of the KR, that is the preservation of the monotonic extension principle.

Divergent knowledge management covers those services through which people can express reaction and in consequence reveal knowledge divergence. This component it is embedded in the ks-workspace. It imposes a representation of the discussion that takes place and which is modelled through the discussion thread model. The discussion tread proposes a series of primitives, each one to model the different kind of motivations, and a set the pre-established associations to combine them.

The awareness component is in charge of facilitating internalization. This component keeps people up-to-date about knowledge evolution. Awareness mechanism is specialized in: a knowledge awareness mechanism that is a change awareness mechanisms with the characteristic of having an explicit representation of the knowledge and a discussion awareness mechanism that is part of the previous one with special focus on the discussion thread behavior.

The remainder of this chapter will deal with ks-workspace component and knowledge divergence management, leaving awareness services for the chapter 5. This chapter is organized as following. The section 5.2.1 presents the knowledge artifact concept as the minimal unit of knowledge. Then, the section 3.3 deals with knowledge-sharing services on the top of the workspace. Externalization and publication service are discussed in a general way, independent from the knowledge representation system. It is suggested a shared workspace with two functionalities: private and shared knowledge repository manipulation. They are the private and shared workspace. The section 3.4 deals with the augmentative development of the KR. Finally, I will complete this approach by presenting the discussion activity support that allows the occurrence of knowledge divergence. The discussion thread component is presented as the resource to manipulate discussions.

3.2 Knowledge Artifact

A unit of knowledge is what was previously called, in Chapter 2, section 2.3.3, the *knowledge artifact* (k-artifact). Thus, k-artefact describe the minimal unit of "explicit" knowledge. Depending on the level of formalization of the knowledge representation system, a knowledge artifact can be: *informal*, where knowledge is strong hard-coded (i.e. a document, the text of the body of an email or an instant message); or *semi-formal* where informal knowledge representation is mixed with formal representation (i.e a text document classified through a more formal system as in OntoShare [Davies03] where documents are classified by domain ontologies or typed messages like in Aulanet [Gerosa01] or WebGuide [Stahl99]); or *formal* where knowledge is represented by a formal knowledge representation system (i.e. by means of a domain ontology). In case of formal k-artefact, the own artifact is a formal representation of a knowledge unit in terms of the primitives of the knowledge representation formalism. For example, the simplest k-artifact represented by ontologies may be a class, or relationship or an instance. In Chapter 4, section 4.2, where I suggest ontologies as a knowledge representation system, ontological knowledge artefact are detailed.

Independently of the knowledge representation formalism that is used to convert tacit knowledge in explicit, it is possible to propose a general design of a the KR through the *k-artefact*. In general terms, it is possible to see a KR as a collection of k-artefact. However,

depending on the level of the formalization of the k-artifact, these collection are arranged in a particular way. For example, if the the knowledge artefact are emails they can be as a chronological ordered collections of mails, while in a semi-formal, a collection of typed emails can be ordered according to the subjects and type. Formal k-artefact are ordered according to the rule of combination of the primitives of the knowledge representation paradigm.

Knowledge artefact, therefore, allow one to make the knowledge-sharing workspace independent of the level of formalization of the knowledge. In the remainder of this chapter I will manage the k-artefact without presupposing any knowledge sharing formalism.

Knowledge artefact are also useful to identify the unit of exchanged knowledge. That is, the k-artifact identifies the unit of knowledge that was externalized, and in a publication, the k-artifact is the subject of the contribution. Even, in a discussion, it identifies the knowledge unit to be subject of the reflection of the group.

Speaking of the discussion activity, k-artefact can also help to capture the essence of those contribution by reaction. Later in section 3.5.1, "discussion artifact" will be introduced as the k-artifact that participates in a discussion and allows expressing knowledge divergences.

3.3 The Knowledge-Sharing Workspace

The knowledge-sharing workspace (ks-workspace) has to be a shared workspace that supports the collaborative development of a KR. This development is carried out through the knowledge conversion proposed by ks-process. Therefore, the ks-workspace has to provide the mechanisms that enable both individual knowledge and private activity, and shared knowledge and public activity, because:

- it should host the shared knowledge context as a shared KR
- it should host each individual knowledge context as a private KR
- it should allow knowledge conversion from:
 - tacit-to-explicit knowledge conversion, and
 - private-to-shared knowledge conversion

Besides, the ks-workspace has to assist people to:

- support private and public actions in a differentiated fashion,
- alternate between shared and private KR
- allow exchanging knowledge between both knowledge repositories

Therefore, it is possible to notice that a ks-workspace does not only host the KR (both, individual and common understanding), but also it defines a set of allowed actions which can be executed privately and publicly. In order to achieve this requirement, I suggest

that the ks-workspace will be made up of public workspace and a private workspace to represent the individual knowledge repositories and the private activity, and the shared KR and the public activity respectively. I call them the public and the private workspace respectively. This approach is aligned with the ideas of Pinto et al. that claim to preserve individual privacy for supporting distributed, loosely-controlled and evolving engineering of ontologies (DILIGENT) [Pinto04]. Figure 3.1 shows a conceptual schema of the appearance of the ks-workspace, where rectangles identify the different workspaces. The central workspace represents the public one. In this figure, it is also possible to observe that each workspace hosts its own KR, which can differ among them. The notation ka_i represent a k-artifact. Particularly, the k-artifact ka has been externalized by the participant A and then, it has been published.

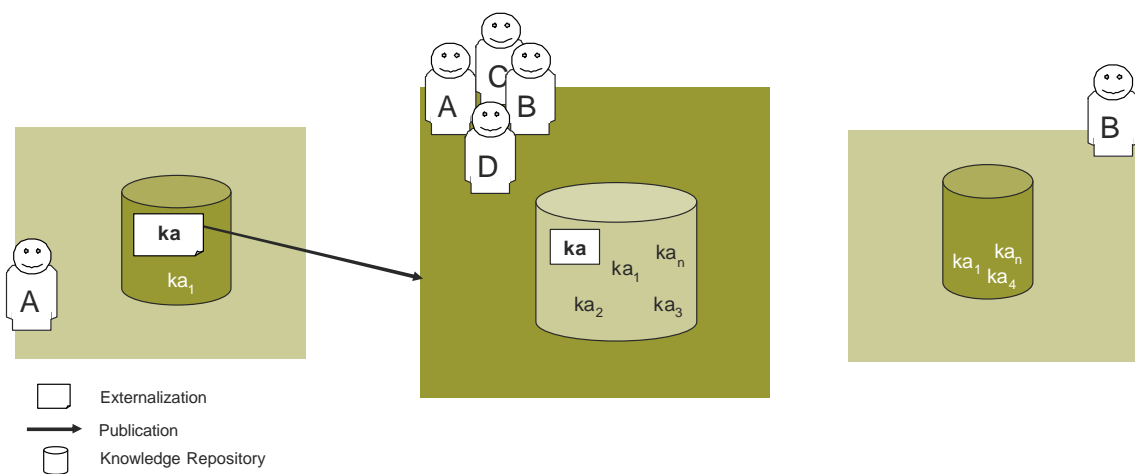


Figure 3.1: A schema of the ks-workspace

The remainder of this section is in charge of providing with operative details of the ks-workspace. They are: the management of their private and public versions of the KR, the private and public activity, and finally the private and public workspaces.

3.3.1 The Management of Knowledge Repository Versions

The KR plays the role of the shared artifact in the collaborative process. It is developed through member's contributions, more specifically through externalization and publication of the knowledge (independently whether the contributions are by reaction). Therefore, from the community's point of view, it has to be the repository of the community's shared knowledge, which represents the *shared knowledge context*; that is, the community common understanding about the domain of interest. However, I have already mentioned that despite the fact that individuals participate in a collaborative activity, they also preserve their own knowledge context. This context is the *individual knowledge context*, which, as opposed to the shared knowledge context, represents the individual understanding of the domain of interest. Therefore, if one understands a KR as the whole externalized knowledge, one can easily get to see the KR of ks-community as the aggregation of many

knowledge repositories. Where there is an *individual knowledge repository* (individual-KR) for each community member and only one *shared knowledge repository* (shared-KR) for the community. The shared-KR represents the externalization of the shared knowledge context, while an individual-KR represents the externalization of the individual knowledge context. However, the shared knowledge context is only made up of those published knowledge.

The *individual-KR* allows only private activities, while the *shared-KR* allows public activities. Externalization action is a private activity through which individual knowledge becomes explicit, consequently, a k-artifact appears as part of the individual-KR. This is the case of the ka k-artifact that is in the workspace of the user A.

On the other hand, the shared-KR is made up of the all published k-artefact and describes the public activities, for example, publication is a public activity by means of a private knowledge artifact becomes part of the shared-KR. Observing the Figure 3.1, this is the case when the ka k-artifact is submitted to the shared-KR.

Moreover, as externalization is a private act and publication is a voluntary one, the shared-KR will only hold knowledge that has been externalized and voluntarily shared. Motivation of remaining externalized knowledge without publication can respond to different private reasons, for example, because the externalization is not ready to be published, or even to express individual interest or because it expresses other perspectives or points of view of the shared knowledge such as knowledge divergence. For example, it is possible to observe in Figure 3.1 that the ka_4 k-artifact remains unpublished in the KR of the user B.

Although, individual knowledge is not part of public knowledge, shared knowledge can be part of personal knowledge. Individual knowledge may also be articulated with a personal view of the shared knowledge (i.e. customized to personal interest). The user B in the Figure 3.1, for example, shares the ka_1 and ka_n k-artefact with the shared-KR.

Managing private and shared knowledge repositories has a direct consequence: there are many knowledge versions that coexist. If one focuses this analysis on an individual, s/he clearly manages two knowledge versions, his/her private version and the shared version. Therefore, individual and shared repositories can be considered as two different versions that coexist at the ks-workspace. They may differ because:

- the individual-KR may not be included in the shared one. The individual-KR contains unpublished k-artefact.
- although they may share a set of k-artefact, both versions may have a different perspective of a shared k-artifact. It is a natural consequence of the differences between the individual knowledge context and the shared one, even these differences are more emphasized among individual knowledge context.

For example, in a KR of documents, individual and shared knowledge repositories can share some documents, each one has its own document, but they can also have different versions of a certain document.

In section 3.4 version differences will be considered as a problem of the collaborative development of the KR and they will deal with techniques of coordination and articulation

of versions. Then, in section 3.5, they will deal with version differences as a way to manage knowledge divergence occurrences.

3.3.2 Private and Public Workspaces

The *Private Workspace* (PrW) is a non-shared workspace that is only accessible by its owner. It hosts the individual-KR and preserves the privacy. PrW supports the private actions. Each community member has a private knowledge space. Externalization is the main allowed activity at the private workspace.

While a participant works only at private workspace, collaboration does not occurs, it can be seen as a single-user application, where each individual can develop a private, personal and individual-KR.

Private workspace gains relevance when individual makes a publication. Publication activity is initialized at the private workspace and finishes at the public workspace.

The *Public Workspace* (PuW) is a shared workspace that is accessible to any community member. It is unique and it contains the shared-KR.

The PuW supports public actions, it mainly is the target of publishing actions. When a publishing action is performed, the PuW receives a knowledge contribution and has to integrate the corresponding k-artifact to the shared-KR. However, the integration of contributions is not as simple as it seems to be. Independent from the knowledge representation systems that is used; it always requires certain degree of coordination and articulation. It is because the act of publishing is a central activity of a collaborative process. For example, in the case of the simplest system, a repository of document, it is at least necessary to have a mechanism to control file names. It is even more complex when a formal knowledge representation system as ontologies is considered. I will present it in detail in section 3.4.

At public workspace there is no externalization, it is only carried out at private workspace, but the shared-KR is developed through contributions (publishing actions).

There is also another activity that can be considered as inherent to the public workspace, that is not basic of the ks-process, but is basic of this conception of the knowledge sharing workspace: it is the action that allows one to transfer a "view" of the shared-KR to the private workspace. I call this operation: "transference".

Although in this chapter, I only focus on externalization and publication actions, the activities related to reaction and internalization find in this workspace the source of information. For example, later in this chapter, discussion activity will be discussed in the context of the public workspace. And, in Chapter 5, the activity on top of the public workspace will be used as a source of awareness information.

Private and Public Activity

According to the structure of the ks-workspace, the ks-activity has to be considered in the context of the individual and the shared ks-workspace separately. Consequently, ks-activity is reorganized in private and public activity respectively. They are differentiated because whereas the execution of public actions "is perceived" by any community member, private actions are not public; their execution is hidden to the other members. Private

activity defines a set of valid private actions that can be performed over the PrW; whereas public activity defines those valid actions over the PuW. Both, private and public actions come from the ks-process; they only classify externalization and publishing as private and public actions respectively. They also establish the means to manipulate both the private and the shared knowledge repositories.

Private actions Private actions are necessary to execute the private activity. They are performed at the PrW. The main private action is the externalization. *Externalization actions* are the needed actions to make the knowledge explicit; that is, to create and update a k-artifact.

Private actions are dependent on the knowledge representation system. Depending on the knowledge representation system, the externalization activity is specialized in a set of allowed edition actions. In the case of an ontological representation, these actions are those, for example, to edit an ontology. In Chapter 4, it will be shown the actions corresponding to the ontological paradigm.

Public actions are those performed at PuW to execute the public activity. They are the *publishing actions*, that involve all those actions that allow making knowledge contributions. Publication means to make a *contribution* from the PrW to the PuW. A contribution contains a k-artifact coming from the individual-KR.

Publishing actions are the most interesting ones; they focus on publication and reaction ks-process's steps, they take place at the public level (they imply a collaborative activity) and they also involves changes at the shared-KR. In this section, I deal with publishing actions from a general point of view; however they required a more detailed treatment when they are considered as contributions of divergent knowledge. Because publishing actions are not only the actions to make a single contributions, but also to give conflictive contributions or even to publish argumentations. Last approach will be tackled in sections 3.5 and 3.5.1.

There is also another activity that can be considered as public action; it is the action that allows one to make a contribution from the shared-KR to the individual one. It is not a ks-process action, but it is operatively useful. I call this action: *transferring action* and works as a publishing action, but it takes a contribution from the PuW to a PrW. It is responsibility of the user to select which k-artifact to transfer. This action allows defining a "view" of the shared-KR at an individual one.

Other public activities that are complementary to the ks-process, like socialization, may be considered. However, they are out of the scope of this thesis, because they keep the knowledge in a tacit state.

There is a third group of knowledge-sharing actions as well. They are those that allow browsing and querying the KR. They are the *consuming actions*.

Consuming actions are specific actions for retrieving knowledge and can take place at both private and public knowledge repositories. These actions can be categorized into browsing actions and querying actions. *Browsing actions* involve navigating the

structure of the KR; the collection of k-artefact. *Querying actions* allow retrieving knowledge from the KR through the use of a specific querying language. Queries are dependent on the scheme of the KR, and formal knowledge representation guarantees the richness of the query expression and the result. When k-artefact are ontological knowledge artefact, it is possible to ensure a rich query mechanism.

Although, consuming actions are not specific to develop the KR, but they also are interesting to give suitable awareness information; they are another alternative to assist when internalization takes place. Visiting the knowledge space people can realize about the activity and evolution of the knowledge. Both querying and browsing actions are very useful to analyze and understand the intention of some actions. For example a person that browse the shared-KR, may be useful to deduce the individual interest in the visited k-artefact.

To sum up, these knowledge-sharing actions (ks-actions) are the means to execute the ks-process and even the means to consume the stored knowledge. These ks-actions are the right actions through which the KR is manipulated. They can be classified in three main groups: externalizing actions, publishing actions and consuming actions. Externalizing actions are the actions to make explicit the knowledge and it is private; publishing actions are those specific ones to make knowledge contributions and it is public, while consuming actions are those specific ones to retrieve knowledge and can be either private or public. Any ks-action is related to at least one k-artifact. A k-artifact can be consumed, externalized or published.

In figure 3.2 a conceptual model of the ks-actions is shown independently if the ks-actions are private or public. Ks-actions are organized in a *is-a* hierarchy. In this figure, it is also possible to see the relationships that a ks-action states with the member and the k-artifact. The **Member** concept represents the user who has performed the ks-action and the **KnowledgeArtifact** concept represents the k-artifact involved in this action. The diagram presented in the figure has to be interpreted following the semantic of UML [Jacobson99]. Later in section 3.5.1 this conceptual model will be completed by incorporating the actions that come from the discussion activity.

3.4 Augmentative Development of the Knowledge Repository

The collaborative development of the KR is a consequence of a mixture of private and public activity. This is achieved through externalizing knowledge at the individual-KR, and its subsequent submission to the shared-KR. As it has been said before, externalization is the act of defining a k-artifact where knowledge is represented according to the knowledge representation paradigm embedded in the workspace. However, publishing is not a direct action because any knowledge contribution should provoke a consistent shared version, in the sense that the contributed k-artifact could be "integrated" to the shared version without meaning any cognitive conflict. Integration means fitting the contributed k-artifact to the shared version.

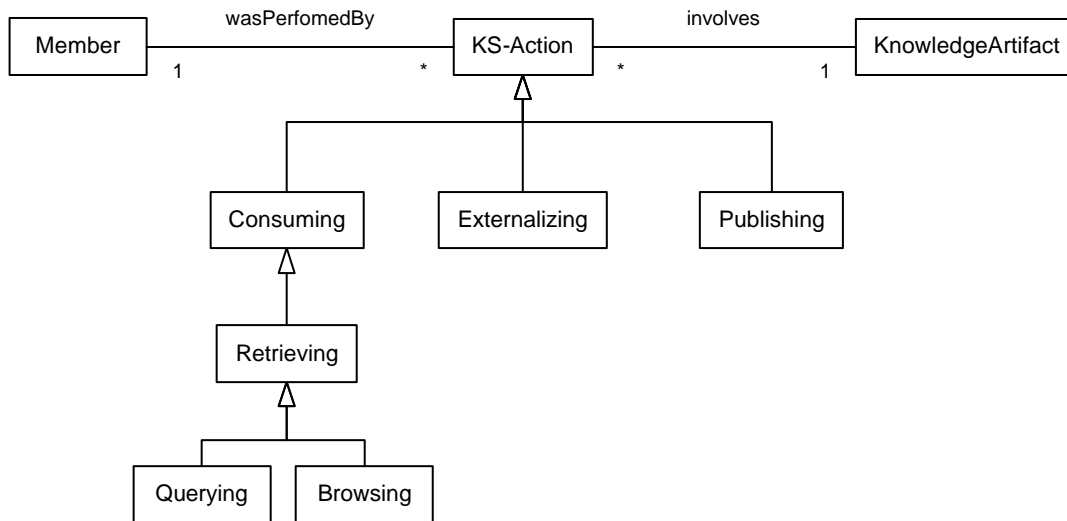


Figure 3.2: KS-actions conceptual model

Detecting the occurrence of this kind of conflictive situations is a requirement for computer-supporting of the collaborative development of the shared-KR. Traditionally, it is a well studied problem in the literature of CSCW. Works of Dourish [Dourish95] and Molli [Molli01] are some examples of them. Dourish characterized this situation as inherent to multi-synchronous applications that are characterized by their abilities to support divergences, i.e. parallel stream of activities on replicated objects. Dourish wrote:

"Working activities proceed in parallel (multiple streams of activity), during which time the participants are disconnected (divergence occurs); and periodically their individual efforts will be integrated (synchronization) in order to achieve a consistent state and progress the activity of the group".

Molli et al. refines it saying that:

"The main difference between synchronous/asynchronous and multi-synchronous applications is modifications visibility and integration. In standard collaborative applications, when one user performs a modification, it is immediately visible by others. In multi-synchronous applications, modifications done by one user are visible by other users, only when she (or he) validates her modifications (commit her changes). A visible change does not imply immediate integration by other activities streams. Concurrent modifications will be integrated when users will decide it."

Although these approaches seem to fit suitable to the situation presented here, the difference is that both Dourish and Molli's approaches are based on all the users who work at the private workspace over a shared object that was checked out from them shared repository and will come back updated, following, for example, the Copy-Modify-Merge Paradigm.

However, the development of a shared-KR is a design activity, where the community develops a common understanding, where knowledge divergences occur and discussion about the divergence should be promoted. Besides, every contribution must guarantee a monotonic extension of the KR; this means, every contribution has to provoke a extended version of the KR and does not have to introduce any cognitive conflict at the shared-

KR. In this approach, I intent to continue with the activity even if commitments are not achieved.

When a new k-artifact has to be integrated to a KR, to recognize the problem of conflict occurrence involves detecting potential *inconsistency between private and public versions*. Characterization of the inconsistency types is strongly dependent on the knowledge representation system, for example, in a formal system an inconsistency involves introducing semantic contradictions; on the other hand, in a repository of documents it may involve having two different versions of a same document without distinguishing the relation between them. Despite the fact that the dependency of the knowledge representation paradigm, it is possible to identify two sources of inconsistencies:

- *updating of the private version*. It is the case where both the shared-KR and the individual one share a k-artifact and this k-artifact is updated in the private version. This produces a new version of this artifact that differs from the one in the shared-KR. This is the simplest way of knowledge divergence occurrence. As this updating occurs as a private activity, users may decide to coexist with the conflict privately. It is a private incompatibility situation. It is not harmful except if this new version of the knowledge artifact in question is intended to be published.
- *updating of the shared version*. It is the case where both the shared-KR and the individual one share a k-artifact and this k-artifact is updated at the shared version. This produces a new version of this artifact that differs from the one at the individual-KR. So that, the private version leaves out of date. Although always the shared version is updated in an augmentative fashion, some private versions can remain inconsistent with it.

Notice that inconsistencies among private versions are not conflictive and can remain hidden in the private knowledge contexts without entailing any problem while they are not published.

Previous situations of inconsistency become conflictive when a publishing or transference actions take place. Any publishing action (similarly to transference) involves an augmentative or conflictive contribution. A contribution is *augmentative* if it can be integrated to the shared-KR without introducing any conflict -it does not produce any inconsistency at the KR level. Therefore, augmentative contributions produce a monotonic extension of the KR. Augmentative publications represent the spirit of a knowledge-sharing community, where people like sharing new ideas, solutions and even problems, in this sense by means of a contribution the "wisdom" of the community always augments. On the other hand, a *conflictive contribution* is a contribution that intent to incorporate a contradiction, tries to negate the exiting knowledge. But by the communitarian spirit, nobody has the privilege to change what other participant has already said, at the most s/he should open a debate. In consequence, contributions that intent to "delete" part of the KR should not be enabled, because it would mean the reduction of the common understanding.

Each time a publishing action takes place, it is necessary to check whether it involves an augmentative contribution. Each knowledge representation system proposes a set of

rules that must be validated to ensure an augmentative contribution. These rules specify the condition to achieve a monotonic extension of the KR in terms of the knowledge representation system. Moreover, the resulting version of integrating a contributed k-artifact is augmentative if it holds every conservation rules. Conservation rules determine whether an edition action is conservative.

Contributions that pass the previous checking can be integrated to the target KR without any inconvenience. On the contrary, *non-augmentative contributions* should be rejected to preserve the monotonic extension of the knowledge sharing repository. Operatively, non-augmentative contributions may be solved in three ways.

- One is by fitting the private version according to the shared one, because the private version has been left out of date. In this way the private divergence disappears.
- The second one is to preserve the divergence between the private and shared versions, but in this way, the divergence is private, so that, it does not provoke any conflict at the shared-KR.
- The third one is to publish this knowledge divergence anyway. However, this last option does guarantee a monotonic extension of the shared-KR, except that the divergence contribution might be augmentative.

The divergence knowledge management component is in charge of deal with these two last cases, when the divergence is explicit and coexists in the ks-workspace. This approach will be introduced in next section.

3.5 Divergent Knowledge Management

In this section, I will discuss divergence occurrences at the ks-workspace level. As consequence the ks-workspace has a private and a public workspace; knowledge divergences can occur in two senses: it can be a private divergence or a public.

Private divergence is a knowledge divergence in the individual-KR with respect to the shared one. It is the simplest one and it is inherent to the individual workspace conception; it preserves the autonomy principle. Any modification of shared knowledge at individual-KR means a divergent externalization in the private knowledge version. One of the advantages of separating private and shared workspace is the direct support of this kind of divergence. Private version can be *in conflict* with the shared version. It is *in conflict* if what is held in the shared version is also held in the private version but it is not held "exactly" as it is in the shared version. This means that, there is a k-artifact that is in both versions but it is different and if it was published, it would not be an augmentative contribution. Conflictive private versions must be analyzed in the context of the knowledge representation system; to give a right meaning of "exactly" in previous sentence. In Chapter 4, for example, the concept of local context is used to determine the difference between two version of a k-artifact.

On the other hand, a *public divergence* is a knowledge divergence at the shared-KR level. Public divergence is due to the publication of a non-augmentative contribution. This

means having different perspectives (divergent versions) of a k-artifact in the shared-KR. However, in the approach of this thesis, a non-augmentative contribution is encapsulated in a "discussion artifact" in order to avoid the violation of the principle of monotonic extension of the shared-KR. Below, in section 3.5.1, the discussion thread model will be introduced as the resource to express public divergence as an augmentative contribution.

3.5.1 Discussion Thread Model

In order to support the dialogue among the community members and to allow the publication of non-augmentative contribution to the KR; I have defined the discussion thread model to capture the knowledge discussion activity with the occurrence of divergence. This model captures the nature of the contributions by the reaction that takes place in the ks-process that allows tackling the collaborative development of the KR. This model was developed to provide a simple yet formal structure for the discussion and exploration of knowledge building –the "wicked" problems. It proposes a language and a representation of the discussion, which is central to the process of tackling this wicked problem. In short, the discussion thread is a model for linking contributions by reaction.

The *discussion thread model* is a conceptual model that identify the matrix of logical elements that allows representing the different kind of contributions by reaction. In the context of the ks-activity, it represents the history of knowledge exchange about a particular subject (k-artifact). It is a sequence of knowledge contributions that links those contributions that are related to a particular k-artifact and during a period of time. The discussion thread primitives imply another classification of contributions, but in this case, contributions are classified according to the kind of reaction which may arise. There are different kinds of contributions by reaction (or *discussion contributions*) according to the role they play in the discussion activity. They are classified in two groups of contributions: *knowledge contributions* and *argumentations*.

Knowledge contribution group is made up of augmentative contribution and divergent contribution. These contributions always provide more knowledge, either complementary or divergent.

- *Augmentative contributions* are contributions that produce an augmentative version of an existing k-artifact. They always add more knowledge to the original one and do not imply any divergence; they tacitly manifest agreement with the original contribution.
- *Divergent contribution* are knowledge contributions that give another perspective of an existing k-artifact. They always have the intention to declare a disagreement with a previous contribution. A divergent contribution is the resource to manifest a cognitive conflict –a knowledge divergence. They allow one to publish a contribution does not check as an augmentative contribution.

Any augmentative or divergent contribution has associated a k-artifact, which is called the discussion artifact. A discussion artifact encapsulates the k-artifact which either complements an existing k-artifact through an augmentative contribution or expresses

an alternative to the existing one through a divergent contribution. These discussion artefact are the *augmentative discussion artifact* and the *divergent discussion artifact* respectively. It is also possible to distinguish a third kind of discussion artifact, the initial k-artifact. This special type of a discussion artifact is in charge of identifying the head of the discussion thread. It is the *initial discussion artifact*.

Discussion artefact are a specialization of the k-artifact which are useful to discuss an existing k-artifact. Particularly, discussion artefact are the resource through which divergent versions can coexist in the same KR. They always allow making an augmentative contribution because the encapsulate the the conflict in the discussion artifact which is published. Figure 3.3 shows a conceptual hierarchy of the discussion artefact as a specialization of a k-artifact.

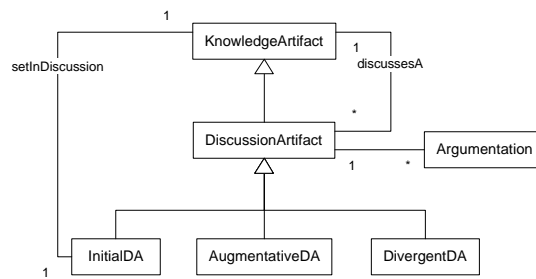


Figure 3.3: Discussion artefact.

The group of *argumentations* allows carrying out negotiation of a conflictive contribution arguing different positions. Argumentations, in general, state an opinion or judgment that either supports or objects to a knowledge contribution. Therefore, they are always attached to some contribution. Argumentative contributions can be sub-classified as *Support* or *Objection* to support or object other thread contributions.

Threads act as the continuous link of the discussion; this means that once a thread has been triggered by an initial contribution, it will be augmented by more discussion contributions. The discussion thread is an aggregation of augmentative and/or conflictive contributions, where *divergent contributions* correspond to different branches in the thread structure. Each branch can be seen as a sub-thread of the original contribution. The discussion thread also holds the argumentations which are attached to the contributions. In Figure 3.4, the conceptual model of the discussion thread can be appreciated.

From the conflict structure point of view, a discussion thread has a tree-like structure as it was possible to appreciate in Figure 2.2 of Chapter 2, where the root represents the initial contribution and each branch represents an augmentative or conflictive contribution in the knowledge discussion. Despite the fact that given discussion thread definition allows one to imagine that more divergence implies a deeper discussion and, in consequence, a deeper tree, it is not so realistic to think that in the real life the thread structure can grow in depth so much, because going in depth in the tree means following the discussion on a subject which does not have consensus.

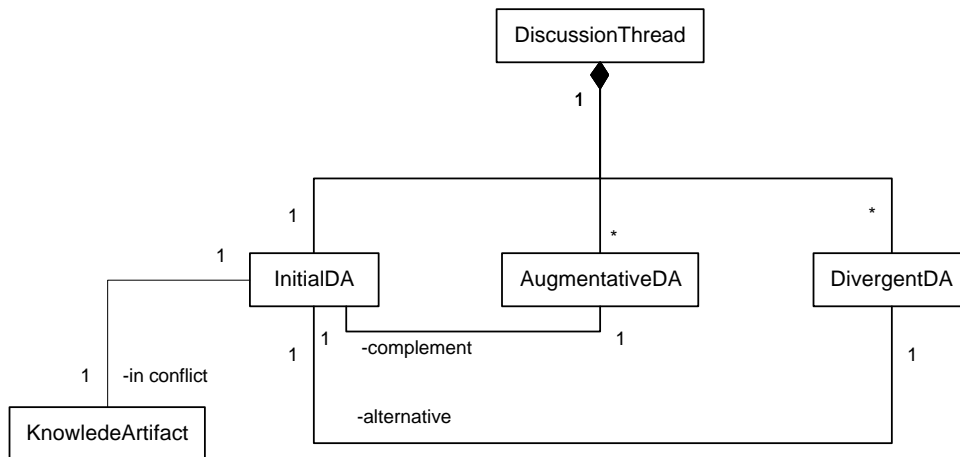


Figure 3.4: Discussion thread conceptual model. It is an aggregations of discussion artefact.

Discussion Thread Development

Any discussion contributions are always attached to an existing k-artifact which has been published by a previous contribution. To trigger the discussion, it is necessary to make the identification of the initial discussion artifact. Then, the linking of the discussion contributions will take place.

Initial contribution identification is a fundamental activity to the divergence occurrence because it allows opening the discussion. It involves the identification of the k-artifact to be set "in conflict". This identified which k-artifact becomes the initial discussion artifact; the head of the discussion thread. It involves specifying what knowledge will be questioned. This is a member responsibility, because users decide what they want to set "in discussion". For example, in a special interest group (SIG) that exchange knowledge by emails, initial contribution identification corresponds to select what email will be replayed. Obviously, email exchange paradigm only supports the email identification, it is impossible to identify the questioned knowledge even if typed emails were used. When it is supported a more formal representation of the knowledge, initial contribution identification becomes more precise as it will be shown in section 4.6.

Divergent contribution linking means attaching to an initial contribution the new discussion contribution typed as divergent contribution. It is also a basic step in the discussion thread development. In the above example of the SIG, contribution linking coincides with the fact of contribution identification because both are embedded in the replay email action. However, in a CKS system, this implies a previous externalization in the individual workspace, and its next publication as a divergent contribution (or alternative conception of the shared knowledge) in the public workspace. Depending on the knowledge representation paradigm these activities have to be refined as it is shown in Chapter 4 when ontologies are manipulated.

Finally, *argumentations* are always part of the discussion thread; this means that they can be linked to any thread contribution. The externalization of an argumentation does not require any knowledge representation formalism, they can be externalized directly in

a natural language.

Discussion Activity

In Chapter 2, discussion activity was presented as part of the ks-activity, where contributions are consequences of reactions, being reaction the ks-process step that enables individual to develop a discussion around a topic of interest. It has also been mentioned that the discussion is the way to tackle design activities, as it is the collaborative development of a shared-KR.

This discussion activity establishes a set of discussion actions. To carry out a suitable discussion activity at the ks-workspace, it is necessary to count with clear identifications of the possible actions that members can perform. Discussion action complements the public actions at the shared workspace. One of the responsibilities of these actions is to allow the development of the discussion thread. To achieve this goal, they are comprised by two groups: the opening discussion action and the discussion contribution actions. In the context of the discussion activity, the k-artefact are discussion artefact.

The opening conflict group covers only one action whose aim is to initiate a discussion around a k-artifact. It corresponds to the initial contribution identification. When a discussion is opened, the existing k-artifact becomes a discussion artifact; in particular, it becomes the head of the discussion thread, representing the k-artifact of the initial contribution. On the other hand, discussion contribution actions allow making augmentative contributions to a discussion thread. This category is specialized in augmentative and conflictive contributions and argumentations. This kind of contributions contribute with a discussion artifact. Figure 3.5 shows a conceptual model of the discussion action hierarchy.

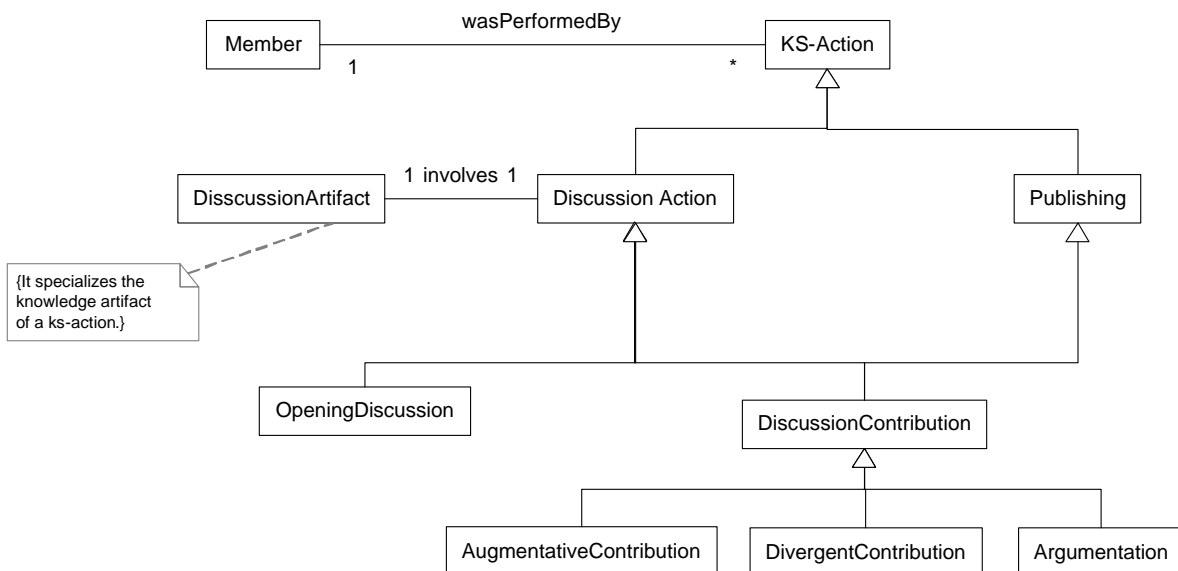


Figure 3.5: Discussion action conceptual model

3.6 Conclusion

In this chapter, I have presented a conceptual framework which models the main component of a CKS system. This framework is based on a ks-workspace that supports the ks-process by means of the community knowledge is converted from tacit to explicit and from private to public. The ks.workspace is made up of two workspaces (the PrW and the PuW) which support the private and public ks-activity respectively.

Besides, in this chapter, the collaborative development of the shared-KR was presented as an augmentative development carried out by the knowledge contributions whatever means a contribution (augmentative or divergent). Then, the divergence occurrence was stated as an augmentative contribution of divergent knowledge. Divergent contributions use discussion artefact to encapsulate the divergent knowledge and are arranged in the discussion thread which models the discussion activity history around a knowledge subject.

Chapter 4

Sharing Knowledge by means of Ontologies

Contents

4.1	Introduction	66
4.2	Knowledge Model: Ontological Artifact	67
4.3	Ontological Representation of the Knowledge Repository	70
4.3.1	Domain Ontology	70
4.3.2	Member Profile Ontology	71
4.3.3	Knowledge-Sharing Action Ontology	72
4.4	Sharing-Knowledge by means of Ontologies	72
4.5	Augmentative Ontological Contributions	75
4.5.1	Checking non occurrence of conceptual description mismatches	77
4.5.2	Ontologies Integration	81
4.6	The Occurrence of Ontological Divergences	81
4.6.1	Ontological Discussion Thread Components	82
4.7	Conclusion	84

This chapter deals with the problem of knowledge sharing by means of ontologies. First of all, I make a brief introduction to the problem of using ontologies as the knowledge representation system. Then, the section 4.2 presents the ontological knowledge model that is, the set of primitives that are considered to represent the knowledge. Next, the section 4.3 deals with the conceptualization of the community's knowledge by means of

ontologies. Consequently, the domain ontology, the user profile ontology and the action ontology are modelled. In section 4.4, it is discussed the ks-activity in terms of ontologies, where it is stated that the ks-activity becomes the collaborative ontology development that respects the monotonic principle and does not introduce divergences (section 4.4). These requirements are tackled with the augmentative ontology development and the divergent ontology development, which are based on the augmentative KR development and the divergent knowledge management, which were introduced in the Chapter 3. Augmentative ontology developing allows making ontological contribution preserving the monotonic principle (section 4.5). On the other hand the divergent ontology development approach guarantees the occurrence of the divergence of the knowledge representation, but it also inhibit the occurrence of inconsistency at the ontology (section 4.6). I suggest discussion thread component as ontological primitives to support divergence without introducing inconsistency at the ontological KR. Readers without know-how about the ontological paradigm, can find a brief theoretical introduction to ontologies in Appendix A.1.

4.1 Introduction

Choosing a knowledge representation system is the first decision to take before facing the implementation of a CKS system. The community needs a knowledge representation system to externalize the knowledge. Knowledge stored in the shared-KR has to be represented following some criteria. This knowledge representation can be informal, semi-formal or formal; going from informal (emails, weblogs, written document) to formal systems in order to develop a formal specification (using ontologies to design a knowledge conceptualization). Every knowledge representation system is based on its own conceptual model. This conceptual model provides primitives to express the knowledge. The conceptual model describes how the knowledge representation has to be understood and it influences in the way of updating the accumulated knowledge.

Although there are different systems to represent the knowledge, in this dissertation I suggest ontologies ([Staab04], [Fensel00]) as the knowledge representation system, because firstly, they provide a shared and common understanding of knowledge in a domain of interest; secondly, they capture and formalize knowledge by connecting human understanding of symbols with their machine processability; and thirdly, because they reduce the ks-activity to the collaborative design of a conceptualization of the domain of interest. Besides, expressing the community' knowledge by ontologies:

- allows having a clear identification of the subject of the k-artifact.
- allows knowledge to be machine processable. It is useful:
 - to facilitate semantic portal creation. Semantic portals rely on domain ontologies to structure and exchange knowledge. [Maedche03]
 - to define mechanisms for browsing and querying the KR
 - to allow reasoning on the top of the KR, and in consequence, making deduction about the community's shared knowledge.

- makes the KR becomes a more portable knowledge.
- allows reusing the KR by means of any system that can understands it.
- allows having even a mixed approach, where formal knowledge (ontologically represented) can be mixed with informal knowledge (document repository)

The decision of using ontologies as a knowledge representation makes me to reconsider the ks-activity and even the knowledge discussion activity, in order to adapt them to the characteristics of the ontologies. For example, now, a knowledge contribution would be thought as an ontological contribution, which gives a formal conceptualization (k-artifact) of a particular knowledge subject. Therefore, the ks-activity may result in the collaborative development of one or many ontologies representing the KR. Besides, this approach must guarantee an augmentative development of the ontological KR (monotonic principle) and the occurrence of divergence of knowledge without causing the occurrence of ontological inconsistency.

Traditional groupware applications to develop an ontology collaboratively does not take into account that divergence can arise when a group is designing an ontology collaboratively. Tools like Protégé [Gennari03] or Ontolingua [Farquhar97] consider collaboration just by the fact that users can access remotely to edit a centered ontology, but they do not care about problems of coordination and synchronization of the edition. These problems are considered out of boundaries of the application. Different version of conceptualization, for example are solved through participants socialization (i. e. in face-to-face or remote meeting). Although this approach can be seen as highly sophisticated by some kind of communities, it is largely useful for those communities that have a strong experience in conceptualizing knowledge domains such as software engineering communities.

4.2 Knowledge Model: Ontological Artifact

The different languages, environments and tools for building ontologies impose a variety of primitives for ontology modelling; however *concepts*, organized in *taxonomies*, *binary relations* and *instances* are the only components that can be represented in all of them [Corcho03]. Therefore, in order to unify the different approaches, I have chosen a particular knowledge model to represent these ontology's primitives. This knowledge model is frame-based: frames are principal building blocks of ontologies. Each frame has a single unique name. This model distinguishes the following types of frames: classes, slots, and instances. Particularly, this is the basic model of Protégé 2000 ([Grosso00], [Gennari03]), the environment that I have used to develop the prototype system, which has a frame-based model⁴.

⁴ Frames are the principal building blocks of a knowledge base. Protégé ontology consists of classes, slots, facets, and axioms. Classes are concepts in the domain of discourse. Slots describe properties or attributes of classes. Facets describe properties of slots. Axioms specify additional constraints. A Protégé-2000 knowledge base includes the ontology and individual instances of classes with specific values for slots.

A *class* is defined as a set of entities. *Instances* of a class are elements of this set (the class of an instance is called its *type*). Classes constitute a taxonomic hierarchy with multiple-inheritance. If a class **B** is a *subclass* of a class **A** (its *superclass*), then every instance of **B** is also an instance of **A**. For example, a class **Publication** can represent a set of all publications. Its subclass, a class **Book** represents books, all of which are also publications. Each instance can have only one type.

Slots are also frames (i.e., slots are first-class objects in our model). When a slot is *attached* to a class (its *domain*), it defines binary relations in which instances of that class can participate in and attributes of the instances. For example, a slot **title** attached to a class **Publication** represents titles of publications. If a slot has multiple domains, then instances of all the domain classes have the slot. A slot can have a *range*, which restricts the values a slot can take. A slot range can be another class (e.g., a range of a slot **author** is the **Person** class), in which case a slot defines a binary relation between an instance of a class and the slot value (i.e., between an instance of **Publication** and an instance of **Person**). A slot range can also be a primitive datatype (e.g., a range of a slot **title** is a *String*). It is defined the following primitive datatypes: **String**, **Integer**, **Float**, and **Boolean**.

Slot values must belong to the defined range of the slot: if the range is a primitive datatype, slot values must have that datatype; if the range is a class, slot values must be instances of that class. In the case of multiple range definitions, I assume union semantics: the value of a slot must be an instance of any of the classes in the slot range.

The number of values that a slot can have for each instance is limited by the slot's *cardinality*. Each slot has a minimum cardinality that defines the minimum number of values a slot must have and a maximum cardinality, which specifies the maximum number of values for a slot. If a maximum cardinality is not defined, the slot can have any number of values.

Slot attachment is inherited from a super-class to its subclasses: a slot attached to a class is also attached to its subclasses. When we attach a slot to a class, its range and cardinality constraints are by default the same as for the frame representing the slot. However, we can further restrict the values locally. For example, suppose a slot **publishedIn**, representing a place where a publication was published, has a range **Publication** (which has such subclasses as **Journal**, **ConferenceProceedings**, and so on). When we attach the slot **publishedIn** to the class **JournalArticle**, we can restrict its range to the class **Journal** (the subclass of the global range **Publication**). Similarly, we can limit the cardinality of a slot locally. Local range and cardinality restrictions are inherited to subclasses of a class. They can be further restricted in subclasses.

To make a diagram of the ontology I will use UML notation [Jacobson99] because it is widely known and besides there are many approaches that show how to translate UML specifications to particular ontology language [Knublauch04]. Mainly, UML static diagrams (class and instance diagrams) are used, but the same approach can be taken to extend to dynamic aspect if they were requested by the knowledge domain. Besides, OCL (Object Constrain Language) will be use in the case a more rigorous specification were required.

Ontology manipulation can happen in two different ways, one at conceptual level, where classes, slots are defined, and the other at concrete level, where instances of the

previous ones are defined and manipulated. In Figure 4.1 the three level of the knowledge model are shown. The upper level represents the metamodel of the modelling primitives, while the other two represent the conceptual and concrete level respectively. The conceptual level allows one to define domain ontology (see section A.1). The concrete level or the level of instances, is also called the *knowledge base*.

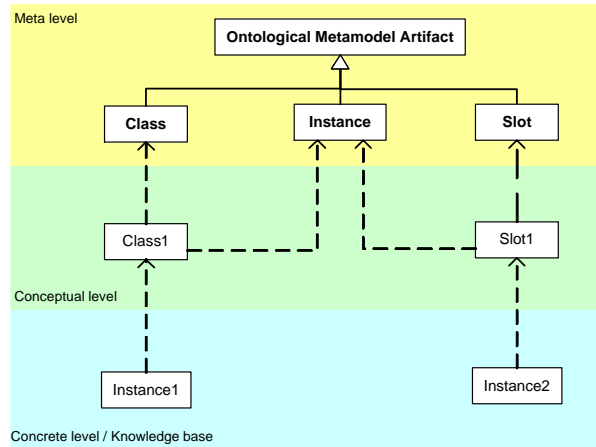


Figure 4.1: UML diagram representing the ontology knowledge model.

Focusing in the ks-activity, ontologies allow one to have a *conceptualization* of the shared knowledge, where any k-artifact is expressed by means of an ontological conceptualization. In particular, any conceptualization is expressed in terms of a set of ontological primitives (in this case, classes, slots and instances). A conceptualization of a k-artifact is called an *ontological knowledge artifact*, or in short *ontological artifact*. Figure 4.2 shows the composition of an ontological knowledge artifact. `OntologicalArtifact` class models a set of `ontological primitives` that are related to an `ontological structure`. Ontological primitives are any element of the conceptual or concrete levels. Ontological structure models the different ways of combining ontological artifact according to the knowledge model.

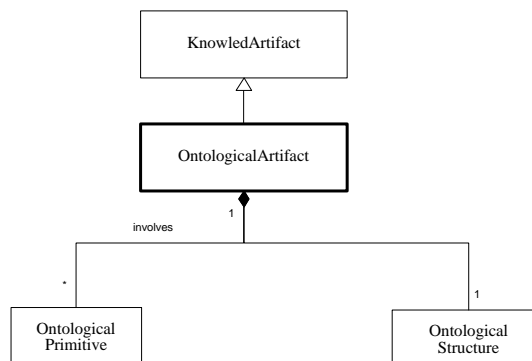


Figure 4.2: The conceptual model of an Ontological Knowledge Artifact.

Having a ontological model of an ontological artifact will be useful for identifying the

k-artifact of an ontological contribution as shown in section 4.5, but also for identifying the subject of a discussion, as it will be discussed in section 4.6.1.

4.3 Ontological Representation of the Knowledge Repository

The goal of this section is to show an ontological representation of the community's knowledge, which was already presented at section 2.3.2. It presents a static view of the KR, focusing on what the knowledge is and how it can be represented by ontologies. Dynamic aspects of the building of the KR (how it is collaboratively developed as consequence of the ks-activity) will be dealt with in sections 4.4.

By taking an ontological approach to represent the community's knowledge, the KR becomes a set of interrelated ontologies where each one represents a certain kind of the knowledge. Therefore, there is an ontology for each kind of shared knowledge and they are: the **domain ontology**, the **member profile ontology**. However, ontologies are used also to model the knowledge about the ks-activity as the knowledge sharing actions and conflict occurrences. This two last cases are represented by special ontologies: **action ontology** and **conflict ontology** respectively. Action ontology will be presented in this section, but conflict ontology will be treated in section 4.6.

Particularly, the **domain ontology** will be developed collaboratively, being the subject of the ks-activity and discussion. Other ontologies are not developed collaboratively, but also they are only pre-established and instantiated by explicit members' actions or derived from their activity, but are not subject of discussion.

In terms of the classification of ontologies given in section A.1, the domain knowledge is modelled through a domain ontology; while the member profile, knowledge sharing action and conflict ontologies respond to a generic ontology type. Besides, the knowledge domain ontology is the only ontology that will be completely manipulated by the community. Nevertheless, the generic ontology has a pre-defined conceptual level specification and they will be populated in based on the participants' activity.

4.3.1 Domain Ontology

As it was introduced in section 2.3.2, the community accumulates knowledge about a domain of expertise or interest. This knowledge is represented by the **domain ontology**. This ontology is build up throughout the ontology primitives (classes, instances, slots). For example, Figure 4.3 shows the conceptual level of the domain ontology corresponding to the scenario presented in Chapter 2 section 2.3.1, and a partial concrete level of it.

This ontology is the core of the shared-KR, and it is the shared object among community's members. The domain ontology is the result of the collaborative processes to build the shared-KR. Participants of the community build the **domain ontology** and can "create and update" it in an augmentative fashion. Next, the section 4.4 will focus on introducing how the community develop the domain ontology in a collaborative fashion.

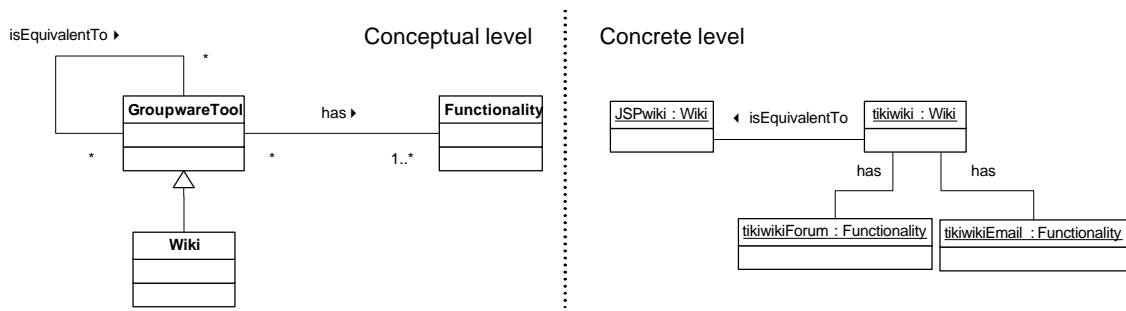


Figure 4.3: Conceptual and concrete representations of the knowledge domain ontology corresponding to the scenario of section 2.3.1

4.3.2 Member Profile Ontology

This ontology describes the knowledge about the community members and their profile. This ontology models social knowledge (filiation, role, etc) and the user profile. The `member profile ontology` enables one to represent the expertise and interest of members in the community knowledge. A member profile is made up of a set of interest and a set of skills. Skills define associations with k-artefact from the knowledge domain and represent the knowledge abilities of the member. In ontological terms, skill defines association between a member and ontological artefact from the domain ontology.

On the other hand, interests are associated to any k-artifact of the KR. For example, someone may be interested in another member or in a particular knowledge sharing action (see `ks-action ontology`). Therefore, interests define associations between a member and ontological artefact from any ontology of the KR (domain, member profile, knowledge sharing action and conflict ontology).

Both skills and interests define association with ontological artefact of the knowledge base (with instances), but in case of the domain-ontology, member profile can also express an interest or skill in a conceptual level ontology artifact; for example in a class or slot. Figure 4.4 shows a simple conceptual representation of the member profile ontology.

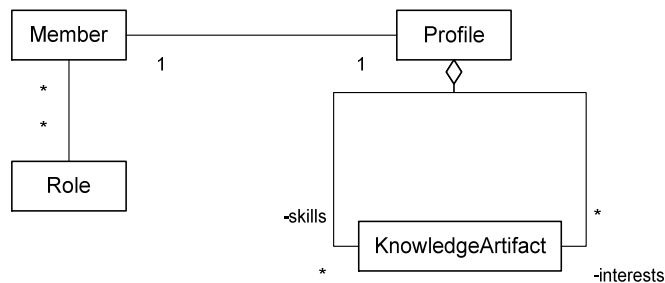


Figure 4.4: The member profile ontology

4.3.3 Knowledge-Sharing Action Ontology

This ontology models the possible action that the community can perform to share knowledge. These actions are those mainly discussed in Chapter 3, section 3.3.2 and 3.5. The knowledge-sharing actions ontology (ks-action ontology) joins both the general KS-action conceptual model and the discussion action conceptual model. In Figure 4.5, any knowledge-sharing action is modelled by the **KS-Action** class. **KS-Action** class is then specialized in **Consuming**, **Externalizing**, **Contributing** and **DiscussionAction** classes. Every knowledge sharing action is related with a user performer (**Member**) and at least, one **KnowledgeArtifact**, which, in case of ontologies, is an ontological artifact.

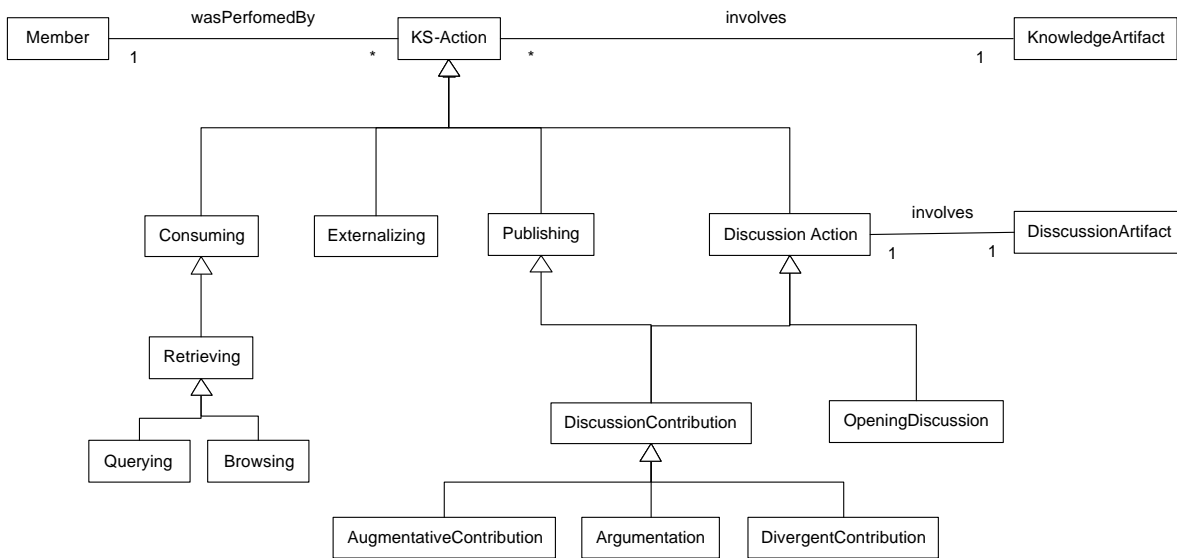


Figure 4.5: KS-action ontology

This conceptualization models a general approach about the knowledge sharing. However, discussion activity requires of a more detailed treatment. Any discussion action is also related to the member and a k-artifact, however in the context of a discussion action, the k-artefact are specialized in discussion artefact. They are useful to model the discussion thread components and will be part of the **conflict ontology** (see section 4.6.1).

4.4 Sharing-Knowledge by means of Ontologies

In section 4.3, it has been already stated that the KR is a set of ontologies that represent the different kind of shared knowledge and that the collaborative activity focuses on developing the **domain ontology** (see 4.3.1). However, this section goes in deep about how the ks-activity is performed when knowledge is represented through ontological paradigm. In order to achieve this goal, the knowledge sharing frameworks, which has been presented in Chapter 3, has to be reconsidered in terms of ontologies. Mainly, the ks-workspace component and the divergent knowledge management have to interpret externalization and publication actions as actions that manipulate an ontological KR.

According to the ks-workspace presented in Chapter 3, people move between both workspaces, the PrW and the PuW, and have two versions of the KR, the individual-KR and the shared-KR. When ontologies are used, the knowledge stored in the knowledge repositories will be a set of ontologies that represent the community's knowledge. Although, every ontology that has been described in section 4.3 should be hosts at the ontological KR, only the **domain ontology** will be considered, because it is the ontology that will be developed collaboratively. The others may not be developed collaboratively.

People need to be able to manage both versions of the domain ontology: the **individual domain ontology** and the **shared domain ontology**, representing the individual and the shared-KR respectively. There is only one **shared domain ontology**, but there are many **individual domain ontologies**, one for each member. Both kinds of domain ontology respect the structure defined in section 4.3.1, but represent different knowledge spaces. Besides, they are developed following different modalities; while **individual domain ontologies** are developed through private action (by externalizing ontological knowledge artifact), **shared domain ontology** is developed by public actions (mainly, by publishing ontological knowledge artefact).

As the externalization step absorbs the impact of using ontology, externalizing knowledge involves building a *conceptualization* of some knowledge subject. This conceptualization is achieved through the direct manipulation of ontology primitives. The resulting conceptualization is an *ontological artifact* (section 4.2). Making externalization by ontologies requires the specific tools to edit ontologies in the PrW, like that used in the prototype presented in Chapter 6.

In this context, publication means contributing to the shared-KR with a ontological artifact. Being the shared-KR the **shared domain ontology**, the publishing action involves to "integrate" to this ontology an ontological artifact coming from one **individual domain ontology**. This contribution to the **shared domain ontology** will be called an *ontological contribution*.

The ontological ks-workspace conserves the characteristics, which were described in section 3.3, except that now the ks-workspace has ontological representations of the shared and the private knowledge repositories. The ontology edition is carried out at PrW and then the resulting ontological artifact is published to the PuW. However, to preserve the characteristics of ks-framework, an ontological contribution to the shared domain ontology must guarantee that:

- the monotonic KR extension presented in section 3.4 is regarded. Any ontological contribution must result in an augmentative extension of the shared domain ontology without introducing any inconsistency. Only augmentative ontological contributions must be allowed; and
- divergence occurrences must be guaranteed. Therefore, it is also necessary to give account with a mechanism that allows expressing divergence with the current shared conceptualization of the domain (**domain ontology**) in an augmentative and consistent fashion. At the domain ontology, divergent ontological contribution will be augmentative and consistent if the underlined model provides the primitives for expressing them.

Let see an example to understand how the modality of collaborative work, which is derived from the ks-framework, fits to the collaborative development of an ontological KR. In Figure 4.6 one can observe a scheme of this example. Initially, the shared version defines concept *a*, *b*, *d*, and *e* as classes and hold them organized in hierarchy. The ontological class *c* will be appear before. The user A's private version is a partial view of the shared ontology, which originally only holds concept *a*, *b* and *d*. But then, user A externalizes the ontological artifact *c* as subclass of the class *b*. Immediately, user A publishes *c* and it is integrated to the shared ontology. The *c* class can be aggregated to the shared versions without any problem, because both the user A's version and the shared version are partially consistent. Then, shared ontology version has changed incorporating the ontological artifact *c*. On the other hand, if at the same time user C is working in its own private version, which contains a partial view of the shared version, and also defines class *c* as part of its private version, a later publication would be a conflictive situation, because s/he has defined the class *c* as subclass of the class *a*. While user C does not make public its externalization there are no problems, because although we can observe a difference among the three versions, this difference remain hidden. By contrary, if user C would want to contribute with its class *c*, it is clear that it brings out a problem of which versions of the class *c* will remain in the ontological repository.

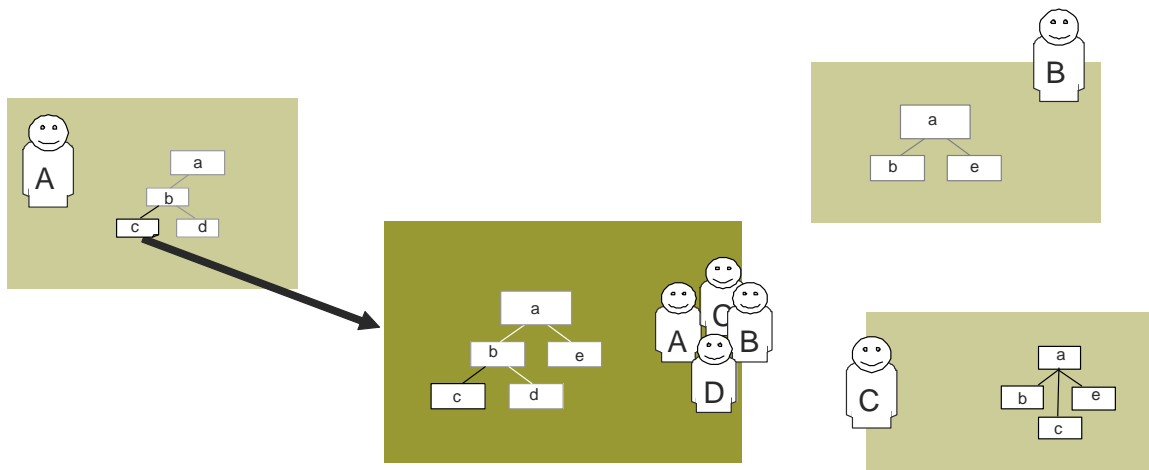


Figure 4.6: Sharing Knowledge by means of Ontologies

An ontological contribution always involves integrating an ontological artifact, which is held at the **individual domain ontology**, to the **shared domain ontology**. However, this integration is not a direct action, because sometimes it may cause some kind of inconsistency at the shared domain ontology or violate the monotonic principle. Many approaches allow integrating user C's perspective; they are:

- *supporting inconsistency management at the shared domain ontology*: Although this inconsistency is the result of allowing the coexistence of multiple conceptualizations of the same subject, it provokes serious problems at the moment of using the knowledge base to make deductions, queries or others.

- *allowing direct updating of the shared domain ontology*: In traditional collaborative approach, where the focus is in the synchronous access to the shared object, user **C** may update the existing conceptualization by incorporating her/his point of view. But, this approach clearly violates the monotonic principle that guarantees an augmentative development of the ontological KR.
- *Versioning management* is another traditional approach for maintaining different perspectives, where the versions represent successive changes at the same "conceptualization" (ontological artifact). Although, it is more suitable to record all members' perspectives, these remain disaggregated.

However, these approaches are deficient to face the problem of developing the shared domain ontology respecting the monotonic principle and simultaneously allowing divergence occurrence. There is a need of suitable mechanisms to integrate all perspectives to represent the coexistence of divergent ones.

Next two sections will be dedicated to introduce the problem of publishing augmentative ontological contribution and divergent ontological contribution. Section 4.5 refers to the augmentative **domain ontology** development and suggests a checking-mechanism to guarantee augmentative contribution occurrences. On the other hand, section 4.6 presents an adapted version of the divergent knowledge management component (section 3.5). It is an alternative approach to introduce divergence at the domain ontology without violating the monotonic principle and without introducing inconsistency; it is achieved by defining discussion thread components as primitives of the knowledge model presented in section 4.2.

4.5 Augmentative Ontological Contributions

An ontological contribution is augmentative if its publication must conserve the monotonic principle enunciated in Chapter 3, section 3.4. To guarantee this, the integration of the ontological contribution to a **shared domain ontology** must be consistent and coherent. That is, it should avoid the occurrence of any ontological mismatch.

Understanding an ontological artifact as an ontology, because it is made up of a set of ontological primitives which are arranged according to an ontological structure, the problem of publishing an ontological contribution to the **shared domain ontology** is reduced to combine both ontologies, the ontological artifact and the **shared domain ontology**. This combination can be done by integrating both ontologies, which means that they are merged into one "new version" of the **shared domain ontology** [Pinto99].

In general, the merge of two ontologies involves making the "alignment" of two ontologies bringing them into mutual agreement, that is, making the resulting ontology consistent and coherent. The problems that underlie the difficulties in merging and aligning are the mismatches that may exist between separate ontologies. Mismatches between ontologies are the key types of problems hinder the combined use of independently developed ontologies. In [Klein01], Michael Klein put forward how the ontologies can differ. Mainly, he states two levels of mismatching: language or meta-model level, and ontology or model level. The former level describes the mismatch between the mechanisms to define classes,

relations and so on. The second level, state the mismatches as the difference in the way the domain is modelled. To tackle language difference, commonly, are taken the approach of translating all ontologies to a common language; Chalupsky's paper [Chalupsky00] is a good starting point about this approach. Although contribution publication may yield language problems, in the context of supporting the ks-activity it is natural to think that the underlying environment imposes a common language (the community share the syntax, logical representation, semantic of primitives and language expressiveness). Therefore, I will concentrate on the problem of difference at the knowledge conceptualization (modelling) that is an ontological mismatch. Following Klein framework [Klein01], I classify the causes of ontological contribution mismatches as *concept description* mismatches.

Concept description mismatches is an *explanation* mismatch because it states a difference in the way the conceptualization is specified. This type of differences are called *modelling conventions* in [Chalupsky00]. Several choices can be made for the modelling of concepts in the ontology. For example, distinctions between two classes can be modelled using a qualifying attribute or by introducing a separate class. These choices are sometimes influenced by the intended inference system. Another choice in concept descriptions is the way in which the **is-a** hierarchy is built; distinctions between features can be made higher or lower in the hierarchy. For example, consider the place where the distinction between scientific and non-scientific publications is made: a dissertation can be modelled as dissertation, that is a book, that is a scientific publication, that is a publication, or as dissertation, that is a scientific book, that is a book, that is a publication, or even as a subclass of both book and scientific publication. Similar situations may arise when other ontology primitives are considered.

Consequently, an alignment between two ontologies is possible if the alignment or the merging of both ontologies does not arise in conceptual description mismatches. Successful combinations involve augmentative ontology development. The augmentative ontology development is achieved if it is possible to make the alignment of an ontological contribution coming from a **private** ontology domain with the **shared domain** ontology without any conceptual structure mismatch occurrence.

Next, I will present a heuristic approach to guarantee augmentative ontologies combination. In particular, the cases of concept description mismatches will be introduced in order to check the viability of a contribution.

To make the analysis of possible concept description mismatches easier, I will consider an ontological artifact as its simplest expression, that is, when it involves only one ontological primitive (Figure 4.7), then, an ontological contribution remains to the publication a simple ontological primitive to the **shared domain ontology**. Therefore, an ontological contribution involves the *merging* of the ontological artifact oa , which comes from an **individual domain ontology** denoted by DO_p with the **shared domain ontology** denoted by DO_s , as long it can be aligned to the DO_s .

By simplicity, in the remainder of this chapter, I will refer to the **individual domain ontology** as the source ontology denoted as O_s , and to the **shared domain ontology** as the target ontology denoted as O_t . In general, it is needed to make the merging of an ontological artifact oa that exists at the O_s with the O_t . More in detail, this involves the merging of the ontological primitive which is referenced by the ontological artifact oa with the O_t . The merging or integration of an ontological artifact oa , involves updating O_t

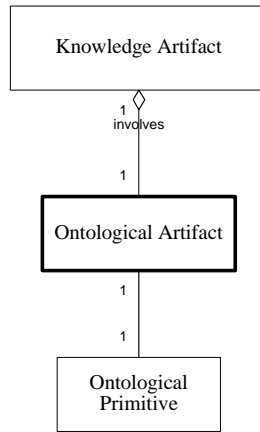


Figure 4.7: A simple ontological artifact

by adding oa , where ontologies \mathcal{O}_t and \mathcal{O}_s may have overlapping parts. This overlapping between both ontologies can be innocuous or it eventually can cause that conceptual description mismatches arise. In the last case, alignment it is not possible. Therefore, to merge both ontologies it is necessary:

- first, to check the viability of the integration, the non occurrence of conceptual description mismatches at the eventual integration of the ontological artifact oa to the \mathcal{O}_t , and
- secondly, to align the ontological artifact oa to the \mathcal{O}_t . Aligning two ontologies implies updating the \mathcal{O}_t by adding oa . As a result, there is a new version of \mathcal{O}_t

Next section (4.5.1) presents a heuristic approach to checking the no occurrence of conceptual description mismatches. Finally, section 4.5.2 details how the alignment of the ontology contribution to the shared domain ontology is done.

4.5.1 Checking non occurrence of conceptual description mismatches

The ontological contribution of the ontological artifact oa to a target ontology \mathcal{O}_t is an *augmentative contribution*; if the integration of oa to \mathcal{O}_t does not provoke any *concept description mismatch*. Previous remark guarantees that the integration of the ontological contribution oa provokes a new version of \mathcal{O}_t denoted as \mathcal{O}_t' where \mathcal{O}_t' is a monotonic extension of \mathcal{O}_t . The possible concept description mismatches are caused by violating the following statements:

- **Conservation of local context.** To integrate an ontological artifact oa to the target ontology \mathcal{O}_t , this ontology must preserve/reproduce the *local context* of the oa in the source ontology \mathcal{O}_s . On the other hand, a local context mismatches occurs. A heuristic algorithm that detects a structural coincidence between the ontological artifact's "context" at the source ontology and the target one is needed.

- **Ontological artifact are unique.** According to the knowledge model, ontological artefact can be distinguished by their identification; the ontological artifact's name, for example. A name mismatch occurs when the contributed ontological artifact has a name that already exists in the shared ontology. If the ontological contribution is a slot called `location` and there is also a class called `Location` at the shared domain ontology, there will be a name mismatch. However in case both ontological artefact have the same type, it may not state a potential conflict contribution, but it may need an updating contribution (next, at the end of this section 4.5.1 details about updating contribution will be introduced).

Checking Local Context Conservation.

Local context is dependent on the type of ontological artifact. According to the knowledge model there are one local context for each type of ontological primitives, that is a *class local context*, *slot local context* and *instance local context*. Let `oa` an ontological artifact that belongs to the ontology `O`, the operation `oa.localContext(O)` denotes the local context of the `oa` in the ontology `O`. The local context of a particular ontological artifact `oa` is a set made up of every ontological artifact related to `oa`.

To enumerate the components of the local context of a given ontological artifact, the conceptual level of an ontology has to be understood as a two-layer graph as is shown in the Figure 4.8 (left side), where in the two layers classes are nodes, but both layers are distinguished according to the type of edges: slot edge or `is_a` edge. Slots edges in the graph connect the nodes representing their respective domains and ranges. `Is_a` edges are useful to represent class taxonomies with multiple-inheritance. On the other hand, the concrete level of an ontology (Figure 4.8, right side) is seen as a simple graph where class instances are nodes, but also class and slot types are nodes; however there are two types of edges, one to identify instantiation (dashed lines) and the other one to represent relationships among class instances, which are instances of slot types.

To make the integration of the ontological artifact `oa`, this approach allows one to look at a very small portion of the **target ontology**; because there is no need of traversing more than one or two links to establish the local context of `oa`.

According to each type of ontological artifact: Class, Slots and Instances, the local contexts are defined. Local context are specified through OCL's `context` primitive, since it enables to define new ontological artifact properties at the knowledge model level. Every local context are defined as follow:

Class local context is made up of all its super-classes and its attached slots, then

```
context Class
def: localContext():Set = self.generalization.parent union(self.slots)
```

where, `self.generalization.parent` denotes the set of super-classes of the current class, and `self.slots` denotes the set of slots attached to the current class.

Slot local context is defined by its value type, then

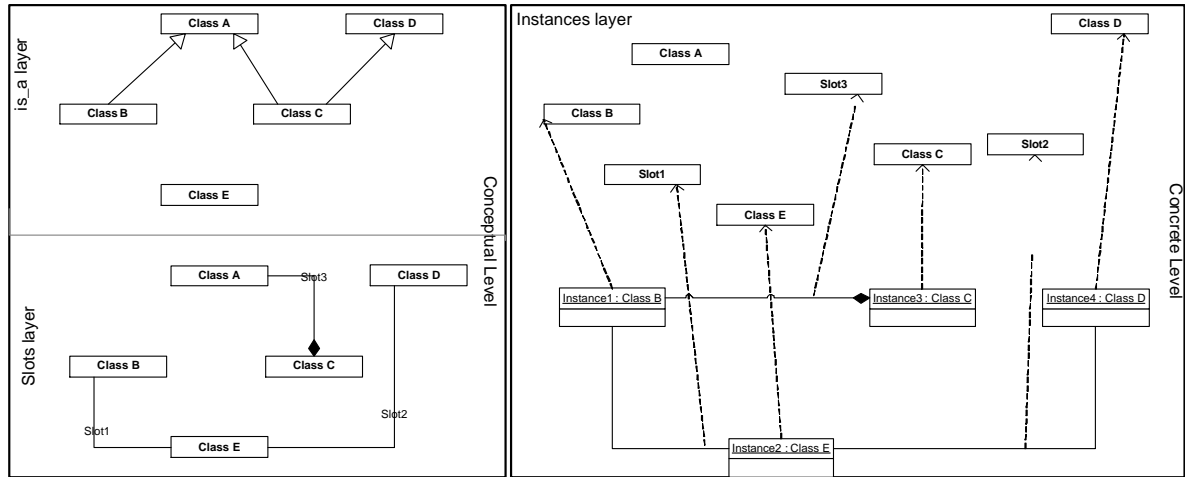


Figure 4.8: Two-layers of an ontology

```

context Slot
def: localContext(): Set = self.typeValues

```

Instance local context is defined by its class and every slots' instances.

```

context Instance
def: localContext(): Set = self.slotValues ->including(self.kindOf)

```

where, `self.slotValues` denotes the set of instantiated slots attached to the current instance, and `self.kindOf` denotes the class of current instance.

Finally, an ontological artifact oa that belongs to an ontology O_s can be integrated to another ontology O_t , if the `oa.localContext(O_s)` has an *image* in the O_t ontology. The image is valid between ontological artefact of the same type and depends on the type of the ontological artifact. The *image* for each type of ontological artifact has been defined as follow:

Class imaging A class C_1 that belongs to the ontology O_1 is image of another class C_2 which belongs to the ontology O_2 if:

1. $C_1.name = C_2.name$,
2. $C_1.generalization.parent = C_2.generalization.parent$ and
3. $C_2.slots$ are image of $C_1.slots$.

Class imaging is only considered at conceptual level, without taking care of instances because in the knowledge model classes are considered as type of the instances (and not as set of instances).

Slot imaging A slot S_1 that belongs to the ontology O_1 is an image of another slot S_2 that belongs to the ontology O_2 if

1. $S_1.name = S_2.name$,
2. $S_1.cardinality = S_2.cardinality$ and
3. $S_2.typeValues$ are image of $S_1.typeValues$.

where $self.typeValues$ denotes the set of $typeValues$ attached to the slot S .

Instance imaging An instance I_1 that belongs to the ontology O_1 is image of another instance I_2 that belongs the ontology O_2 if

1. $I_1.kindOf$ is image of $I_2.kindOf$, and
2. $I_2.slotValues$ is image of $I_1.slotValues$.

In case of instances, the uniqueness of the ontological artifact's name is not possible to check because instances do not posses names. Therefore, the contribution of an instance I of the source ontology O_s can be done as long as $I.localContext(O_s)$ checks at the target ontology O_t . Consequently, instance publishing may yield redundant populations of classes at the target ontology.

typeValue imaging If the $typeValue$ is a primitive, it always exists at both ontologies. However, when the $typeValue$ is an ontological artifact oa , an image of oa must exist at the target ontology.

In case the local context was a set of ontological artefact, the image function would be extended to:

Image of a set of ontological artefact. A set of ontological artefact $S-oa_1$ of the ontology O_1 is *image* of another set of ontological artefact $S-oa_2$ of an ontology O_2 if for each ontological artifact oa_2 of $S-oa_2$ exists an ontological artifact oa_1 of $S-oa_1$ where oa_1 is image of oa_2 .

Checking Uniqueness of Ontological Artefact

Previously, I have sated that there are some cases where the uniqueness of ontological artifact may be not held. That may be the case of an updating contribution. An ontological contribution may be either an original contribution or an updating contribution. An updating contribution is a contribution that updates an existing ontological artifact of the target ontology. Any ontological contribution is a potentially updating contribution if, at least, the contributed ontological artifact has the same *name* and *type* of the another existing ontological artifact at the target version. It is potential because the contribution will be a valid updating if it provokes a monotonic extension of the target ontology, otherwise it cannot be a possible contribution.

This involves the merging of the two ontological artefact, where the source ontological artefact is joined to the target ontological artefact. Therefore, an ontological artefact oa_t that belongs to a target ontology O_t can be updated by an ontological artefact oa_s that belongs to a source ontology O_s , if

1. $oa_t.localContext(0_t)$ is image of $oa_s.localContext(0_s)$ and
2. oa_s is image of oa_s

Mismatch Management

Failed ontological combination gives information about how to determine mismatches. Fails can occur due to: *name mismatches* and *local context mismatches*. Name mismatch occurs due to uniqueness name checking fails. Local context mismatch occurs due to ontological artifact imaging checking fails. According to each frame type, mismatches can be refined to give more precise information about the causes. These fail causes are derived from the statement that are checked when an imaging checking is carried out. Even these causes are useful to give an explanation of the fail alignments.

In a software support, failed-alignment information is useful to assist users to take decision to re-enunciate the contribution. This is the approach that I will follow to state the requirement of Co-Protégé, a computer-support for collaborative development of an ontology that support a ks-activity; it will be detailed in Chapter 6. As other computer supports for ontology merging [Noy00b], in Co-Protégé when a contribution can not be added automatically, user's participation is required. For example, in the particular case of name conflict, users can re-enunciate the contribution as an updating contribution. In case of local context mismatches, they help users to understand the difference between her/his private ontology version with the shared one. As a result of this understanding, users can take different decisions, for example s/he can redesign her/his contribution to fix it to the shared version; to decide the no publication of the contribution, or to publish it as a divergent conceptualization. This last case represents the enunciation of a conflict at the shared ontology, and it is introduced next in section 4.6.

4.5.2 Ontologies Integration

Previous checks guarantee that ontological artifact integration can be done easily. In the case of original contribution (it is not an updating contribution) it only involves adding the ontological artifact coming from the source ontology to the target one, and to reproduce its local context. But on the other hand, the updating contribution means to replace the ontological artifact that already exist in the target ontology by its image existing at the source ontology. That is, the integration of an ontological artifact oa that belongs to the source ontology 0_s to the target ontology 0_t consists of developing the $oa.localContext(0_s)$ in the ontology 0_t .

4.6 The Occurrence of Ontological Divergences

In section 4.4, it has been said that to support the ks-activity by means of ontologies, it is not only necessary to guarantee an augmentative development of the domain ontology, but it is also necessary to account with the possibility to express divergence without introducing inconsistency at the domain ontology. Besides it is important to preserve

that any contribution to the shared domain ontology will be augmentative, even if it is a divergent contribution.

Dealing with ontologies, a divergent contribution means to publish or intent to publish an alternative conceptualization of an existing conceptualization at the shared domain ontology. In theoretical terms, members have to be able to publish alternative conceptualizations, but, in practice, the coexistence of the two conceptualizations in the shared domain ontology entails problems of inconsistency. Therefore, it is necessary to propose some approach that allows maintaining the coexistence of ontological divergence and simultaneously avoids the eventual occurrence of inconsistencies at the shared domain ontology.

In this dissertation, in order to tackle this situation, I suggest extending the ontological knowledge model presented in section 4.2 with discussion thread primitives that enable to support divergent contribution in an augmentative fashion without introducing inconsistency. Thus, the discussion thread model is no longer a conceptual model for understanding the dynamic of knowledge discussion activity where divergences occur at the ks-activity 3.5.1, but it becomes part of the set of ontological primitives to conceptualize the knowledge domain.

In the context of the collaborative design of ontologies, discussion thread will be considered as a concrete resource to make explicit the divergence, where its components (discussion artefact) become first order ontological primitives. However, the augmentative discussion artifact is not necessary to be considered as a primitive, because according to the analysis of the previous section, any augmentative contribution can be integrated automatically without forcing users to make it explicit. Therefore, in an ontological approach to share knowledge, augmentative discussion contributions occur implicitly. As a consequence, the awareness mechanisms has to be in charge of the responsibility of keeping people aware of augmentative discussion threads. In this context, the knowledge discussion activity, will be reduced to open a discussion and to publish divergent contributions. Then, these new ontological artefact are *objected ontological artifact*, *alternative ontological artifact* and *argumentation*.

The main advantage of having an ontological representation of the discussion thread is to encapsulate inconsistencies. A divergence at the shared ontology involves enabling the coexistence of two or more inconsistent conceptualizations. However, these inconsistency conceptualizations remain encapsulated in a *alternative ontological artifact*, and thus, ontological inconsistencies disappear. Following, details about the ontological discussion thread components will be introduced.

4.6.1 Ontological Discussion Thread Components

As it has been already mentioned in section 3.5.1 and then in section 4.3.3, the discussion thread development involves two steps: first, to identify the initial discussion artifact to be set in discussion, and then, to complement it with new discussion contributions. In terms of the ontological discussion thread the identification of the initial contribution corresponds to the identification of the conceptualization to be objected or set "in conflict"; and the second step corresponds to attaching an alternative conceptualization (divergent contribution) to the objected one. Giving argumentations is also part of the discussion

activity, but they only help to give comments to support or dissent with some given conceptualization.

The ontological discussion thread identifies three different kinds of ontological artefact: objected ontological artifact, alternative ontological artifact and argumentations.

Objected Ontological Artifact encloses an ontological artifact (a conceptualization) of an ontology that will be set "in conflict". The objected ontological artifact involves an eventual alternative ontological artifact that may be attached to it by a divergent contribution.

Alternative Ontological Artifact is an ontological artifact that encapsulates an conceptualization that will be given as an alternative to a particular objected ontological artifact. Alternative ontological artefact are always attached to an objected ontological artifact, whereas objected ontological artifact cannot have attached alternative ontological artifact.

Argumentations are ontologically represented but its knowledge is informally represented, that is, they are independent from the knowledge representation system.

To open a discussion, users are forced to identify the objected ontological artifact and then, they may express divergent position as alternative ontological artifact. In the example of section 4.4, if user C wishes to publish its conceptualization: "class C is subclass of class A", s/he must open the discussion by setting the conceptualization "class C is subclass of class B" in a objected ontological artifact and attaching to it her/his alternative ontological artifact. Figure 4.9 shows a schema of the resulting ontological discussion thread.

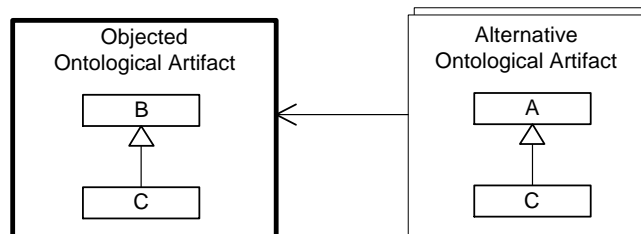


Figure 4.9: An example of a ontological discussion thread

Formally, the ontological discussion thread is an aggregation of an objected ontological artifact and eventually many alternative ontological artefact. The objected ontological artifact is defined by the ontological artifact to be set "in conflict". As the conflict can arise as cause of many different reasons, the objected ontological artifact has also associated a conflict type. Conflict types identify the causes of divergence, for example in the previous example the hierarchical organization of classes C and B is questioned. There are many types of conflicts and they answer to the different kind of conceptual description mismatches that can arise between the objected ontological artifact and the alternative ontological artifact. Figure 4.10 shows the ontological representation of the discussion thread primitives and their relationships. These primitives allow developing the conflict ontology. Conflict ontology completes the set of ontologies that are part of the shared knowledge that were presented in section 4.3.

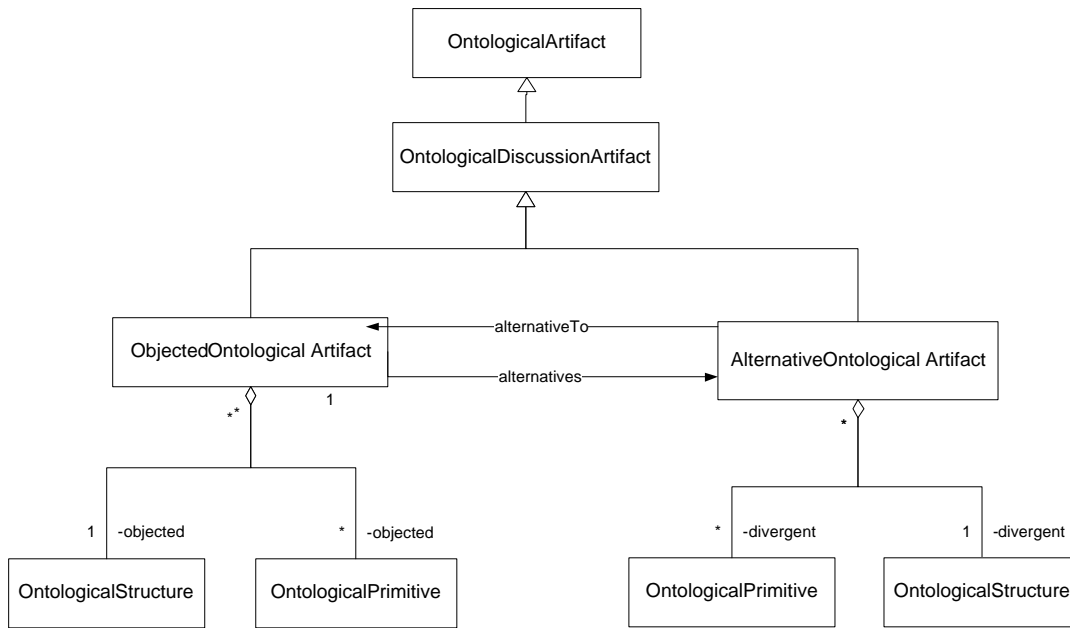


Figure 4.10: The Conflict Ontology

Objected and alternative ontological artefact are two sub-classes of the *OntologicalArtifact* class (section 4.2). *ObjectedOntologicalArtifact* class also models a conceptualization, but in this case the *OntologicalPrimitive* component represents the set of ontological primitives participating in the objected ontological artifact, and the ontological structure is the objected structure. As a set of ontological primitives may participate in different ontological structures, it is necessary to identify which ontological structure is objected. In the case of an objected ontological artifact, the ontological structure component plays the role of conflict type. On the other hand *AlternativeOntologicalArtifact* class represents the alternative ontological artifact given to an existing one (the objected ontological artifact). It also identifies a set of ontological primitives, which are proposed as alternative (alternative ontological primitives) and the way that they are combined (alternative structure). The set of divergent alternative ontological primitives can not necessarily be coincident with the set of objected ontological primitives. Alternative ontological artefact can add new frames to the target ontology. There is no risk of incompatibility because they remain encapsulated in the alternative ontological artifact.

4.7 Conclusion

In this Chapter, it has been presented a concrete application of the ks-frameworks where it was used the ontological parading as the knowledge representation system. Ontologies were chosen because they allow users to design a conceptualization of they domain of interest collaboratively. And thus, the ks-activity was reduced to the collaborative design of an ontological KR.

The ks-framework could be instantiated almost exactly as it was described in Chapter

3, except that the discussion activity was changed a little. Particularly, at the discussion thread level, was possible to remark that in an ontological approach, it is not necessary to have an explicit representation of augmentative contributions because they can be managed implicitly. It is due to the kind of checking that was made to check the occurrence of augmentative contributions.

In this approach, ontologies were not only used to represent the domain knowledge, but also they were used to represent the members' profile and the carried out activity. This last kind of knowledge, which is represented by the ks-action and conflict ontologies, allow becoming the carried out ks-activity as part of knowledge of the community, and thus, they improve quality of the accumulated knowledge. Besides, the fact of having explicit knowledge about ks-activity will allow bringing better support to keep the group aware of this activity.

As a side effect, this instantiation proposes a new approach to carry out a collaborative development of ontologies. This approach is really interesting because allows ontology developers to discuss about the conceptualization. Discussions involve the exchanging of alternative conceptualizations among the ontology developers and the coexistence of divergent conceptualization in the shared ontology.

This approach is in the address of other tools that allows the collaborative design of an ontology like WebOnto [Domingue98] and Apecks [Tennison98]. However, it differs from WebOnto because my approach takes into account the asynchronous development of the discussion and the

Chapter 5

Knowledge and Discussion Awareness

Contents

5.1	The Role of Group Awareness at the KS-Activity	88
5.2	Knowledge Awareness and Discussion Awareness	89
5.2.1	Knowledge Awareness	90
5.2.2	Discussion Awareness	93
5.3	Information Needs	94
5.3.1	Information Needs of Knowledge Awareness	96
5.3.2	Information Needs of Discussion Awareness	102
5.4	Knowledge Awareness Mechanism	109
5.4.1	Gathering of Knowledge Awareness Information	111
5.4.2	Delivering Knowledge Awareness Information	115
5.5	Conclusion	117

This chapter introduces a conceptual framework of knowledge and discussion awareness; the necessary group awareness to support the ks-activity in a CKS system.

Awareness is a relevant component of any groupware application; it keeps users up-to-date about the collaborative activity. According to the nature of the collaborative activity, different kind of awareness can be used. Although, existing awareness types can be applied to the ks-activity, in this dissertation I identify and define knowledge and discussion awareness as the two specific awareness services for CKS systems. While knowledge awareness plays a critical role when it comes to sharing knowledge, since it is not only a means to understand what is going on with the knowledge, but it also becomes a source of knowledge; discussion awareness is in charge of making evident the knowledge divergence occurrences .

This chapter is organized as follow. First, it is presented the role of group awareness in the context of CKS systems. Then, it is introduced the knowledge and discussion awareness as the needed awareness to helps users to carry out the ks-activity. Next, it is developed a conceptual framework which identifies what information should be tracked to provide knowledge and discussion awareness. Finally, implementation features are briefly discussed.

5.1 The Role of Group Awareness at the KS-Activity

In a ks-community, members understand that if they share their own know-how, they can take advantage of other members comments, ideas and points of view, and thus they enrich they own knowledge. But also, if they pay attention to what is going on in the community, they can also take advantage of new knowledge occurrence. These two features are the resource for keeping the community "in action", that is, the community remains continuously engaged in the ks-activity. To keep the community "in action" is always a strong challenger, even though, when it is computer-supported. In this thesis, I have claimed that by means of the support of divergence occurrences in the ks-activity it is possible to improve the conditions of the usability of a CKS system, but it is not enough. People besides need to be aware of the ks-activity to keep them "in action", and mainly they need to be aware of the divergence occurrence to reinforce them.

People unaware of the ks-activity will not be motivated to promote more activity, losing the interest in the community. The fact that people do not realize of the evolution of the ks-activity would entail this activity loses importance. And in a long term, it may be over because the participants have lost interest in an inactive community.

People need to be aware of any new knowledge occurrences in order to achieve a successful ks-activity. They may be aware of ks-activity by browsing and querying the knowledge space finding out the new occurrence of a knowledge contribution; however, it is not enough; this exploratory approach is tedious and hard and, in many cases, it does not reach the news. Or even, they may communicate each other their contributions, but this involves to support the communication by an external mechanism. Users should be aware of ks-activity at any time with the minimal effort.

Therefore, one of the main challenges of a technological support for ks-communities is to stimulate a dynamic activity around the knowledge. This means maintaining a high level of interaction among the community members. It is well studied in the CSCW literature that *group awareness* is the resource to keep the group engaged into the collaborative activity, that is, the awareness which helps to increase the level of interactions ([Dourish92], [Schmidt02]). Then, awareness can be understood as a suitable means to share knowledge, because it can provide information about the ks-activity. Thus, awareness can push the ks-activity. For example, by means of awareness, a user can notice that another user is objecting her/his previous contribution; otherwise, the user needs to manually browse all of her/his contributions to check whether someone has triggered a discussion thread from them.

Awareness should be appropriate to the ks-activity. According to the experience in CSCW field, each collaborative activity may involve reconsidering awareness to make it

more suitable to the activity's needs. In this case, where the collaborative activity is knowledge sharing, it is necessary to identify the own information needs of this activity in order to design a suitable awareness. The main awareness requirement for the ks-activity is to keep users aware of "what is going on with the knowledge". This means that they need information about the occurrence of any ks-activity, which includes mainly the occurrence of knowledge divergences. People need to be aware of the the occurrence of divergences; otherwise, divergence occurrences lose importance in the ks-activity. Therefore, awareness of the ks-activity will be a complement to support divergences in a CKS system.

KS-communities need awareness that will be able of keeping them up-to-date of the ks-activity. As it has been already said in Chapter 3, the knowledge activity includes the ks-actions, where a publishing action is the more interesting because it highlights the moment in which new contributions are made to the community. However, as it also has been discussed in Chapter 3, contributions can be or not part of a discussion thread. To take into account this difference it is relevant to provide a suitable awareness of the knowledge divergence occurrences.

Besides, an effective, proactive and context-sensitive dissemination of awareness information is necessary to really guarantee a more realistic feeling of what is going on in the community. An awareness service will be really effective to keep the community "in action" if the delivered awareness information is necessary and enough to promote the ks-activity occurrence.

The definition of a special kind of awareness for the ks-activity is an advantageous approach to improve community activity. In order to achieve this, in this thesis, I suggest knowledge and discussion awareness as the appropriated awareness for the ks-activity. Knowledge awareness, as it will be discussed in section 5.2.1, is the required awareness to keep the community aware of what is going on with the knowledge. It is beside specialized in discussion awareness, which is the appropriate awareness to follow knowledge divergences, as it is discussed in section 5.2.2. These two kind of awareness are synthesized in conceptual framework which allows identifying awareness information needs in the context of the ks-activity (section 5.3).

5.2 Knowledge Awareness and Discussion Awareness

Knowledge and discussion awareness are new kinds of awareness that focus on the ks-activity which is carried out in a CKS system. Although, other types of awareness as workspace awareness [Gutwin02] or change awareness [Tam04] may be applied in a CKS system, knowledge awareness, are more precise awareness for this kind of activity. Workspace awareness provides "knowledge about what is going on the workspace" and it is more suitable to synchronous interactions and bi-dimensional workspaces, while change awareness helps "to track artefact changes" as asynchronous interaction. Knowledge awareness may profit of both workspace and change awareness, because it may track the asynchronous ks-activity at the PuW and the changes of the shared-KR. According to the approach taken in this thesis, changes are made over the KR by means of contributions. As the KR development is an augmentative development where each contribution provokes a monotonic extension of it, changes to the KR can be independent contributions

or by contributions in the context of a discussion thread, which involves occurrences of divergences.

However, knowledge awareness, as it is asynchronous, definitively differs from another studied types of awareness like awareness of who is around and available for collaboration (e.g. [Dourish92]; [Greenberg96]), and from awareness of clues and turns in verbal conversation (e.g. [Clark96]).

But, helping people to follow discussion thread and to internalize and externalize knowledge is what make knowledge awareness different of these other approaches. Knowledge awareness is favorable to assist users to internalize and externalize knowledge and it is also in charge of keeping the ks-community aware of the knowledge divergence occurrence in the context of the discussion activity that takes place around a k-artifact. And thus, both knowledge and discussion awareness provide the necessary group awareness to keep the community able to understand the collaborative learning process, what means, the knowledge evolution.

5.2.1 Knowledge Awareness

I have defined knowledge awareness as *the needed awareness information to keep a knowledge-sharing community up-to-date about the knowledge evolution* [Diaz03]. Knowledge awareness provides the information that allows individuals to track the ks-activity performed by other participants over the time. It allows answering questions like: *is there a new ks-activity? who is participating in the ks-activity? how has this knowledge evolved?* Keeping people aware of these questions helps to engage them in the ks-activity; because they work as the basic stimulus that a ks-community needs to generate new knowledge; and consequently, it keeps the community "in action". In section 5.3, I will detail all awareness information needs to make people aware of the knowledge activity.

Knowledge awareness, as the means by which individuals track knowledge evolution, is favorable to develop a successful ks-activity because:

- it allows a better understanding of the shared knowledge; because being aware of the place where the activity takes place and which k-artefact are involved, is a helpful way of understanding knowledge conceptualization;
- it induces community participants' curiosity. Curiosity is well known as the key for learning process takes place, and it becomes "the seed" of knowledge internalization,
- it promotes the emergence of "new knowledge". It is a natural consequence of curiosity induction. Members are constantly articulating the received awareness information with her/his private knowledge context and it is the source for the generation of new individual knowledge, which, then, may be contributed, and finally,
- it helps knowledge evolution understanding; along time, it keeps members up-to-date about knowledge progresses through the ks-activity.

Knowledge awareness plays a key role in the ks-process because it works as a means to facilitate the knowledge internalization. Giving suitable information about new knowledge

occurrences (contributions) helps people to notice these occurrences and, consequently, it is a medium to assist people to internalize knowledge. People need clues about the knowledge occurrences to arouse users' curiosity. This is the clue to trigger a learning activity where individuals incorporate this public knowledge into their private knowledge context.

Indirectly, knowledge awareness is also a source of knowledge. Pushing internalization is a way of pushing also the ks-activity, because this internalization becomes the seed of reaction occurrence. Beyond the discussion thread development, when people react, they are providing more knowledge, either augmentative or conflictive, but it is more quantitative knowledge. This additional knowledge emerges as consequence of the stimulus received by the delivered knowledge awareness information. Therefore, knowledge awareness does not only work as a engine of the ks-activity, but also it works indirectly as a source of knowledge.

Besides, assisted internalization is not the only advantage of the knowledge awareness. It can also be useful to aid people to externalize knowledge at their own individual-KR. According to the approach presented in this thesis, where individuals simultaneously manipulate an individual and a shared knowledge repository, knowledge awareness can also be useful to assist people to "externalize public knowledge" in the individual-KR. It implies an indirect externalization at the individual-KR. This means that the new contribution can be automatically incorporated to the individual knowledge context. This feature is mainly based on users' interests. Although, this is a more operational system feature, the lack of it would imply that users have to bring new contributions to his/her individual-KR, turning the updating of the private knowledge tedious. This approach requires of complementary mechanism to notify users about the incorporation of the new k-artifact in its own KR. The notifier component is in charge of giving awareness information about the occurrence of new contributions to the individual-KR. To apply this, it is necessary local awareness that delivers information about changes at the individual knowledge context.

In conclusion, knowledge awareness can deliver either awareness information or concrete knowledge, and both mechanisms are complementary. While awareness information delivering consists of giving information about the occurrence of any event at the PuW; concrete knowledge delivering consists of bringing a knowledge artifact from the PuW to a PrW. The last only applies to knowledge contribution activities. Figure 5.1 shows a scheme where arrows represent the two mechanisms to deliver knowledge awareness.

There exists a third situation that requires of knowledge awareness information. This information is also required to highlight differences between private and shared versions of the both knowledge repositories. As the private version of the KR can be a "view" of the shared version, people needs to be aware of any change at the shared version, in order to tell them that their private version is out-of-date, but changes at the private version must be also tracked to tell users whether their private version is divergent of the public one. This situation has to track changes between both version, the private and the shared ones, and it has to be customized to each user's needs. Although this approach may be partially considered by the indirect externalization, it can be also considered in a independent way; for example, when new externalizations occur in the PrW.

Operatively, the knowledge awareness mechanism of a CKS system has to gather

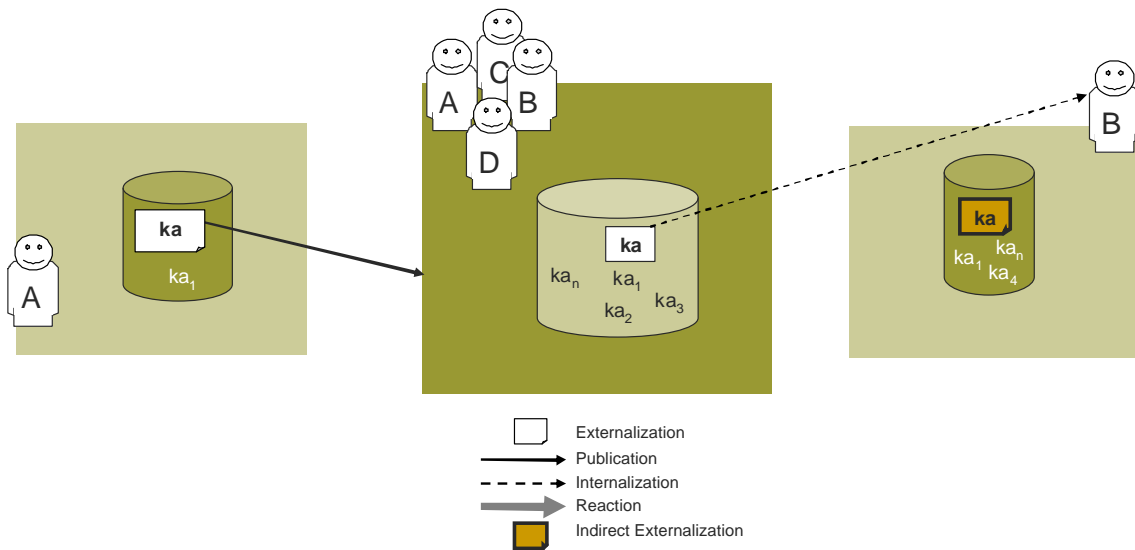


Figure 5.1: Knowledge Awareness as a means of externalizing knowledge

information about the ks-activity in the PuW and then, deliver this information to the users. The gathered information is related to the new knowledge action, this means the awareness mechanism should track both contribution occurrences and consuming actions. People need to track the ks-activity by:

- noticing new knowledge occurrence; although, it is important to track any ks-action, especially knowledge contributions should be tracked since they involves knowledge evolution (changes).
- identifying highly-active knowledge subjects, this means to tracking discussion activity. Any discussion activity must be tracked to understand the discussion evolution, and thus the knowledge evolution. This case involves a particular activity, discussion activity, and it is relevant enough to be dealt as a special kind of knowledge awareness. It is the knowledge awareness of the discussion. In section 5.2.2 it will be deal in details.

On the other hand, delivered knowledge awareness information has to be appropriated to the user needs. Although, in section 5.4 I will come back to these topics and extend them, here, I would want to underlie the importance of the contextualization of the delivered awareness information according to the users needs, in order to achieve the development of a successful ks-activity.

Besides, knowledge awareness may benefit from the knowledge if it is formally represented, because it can be more suitable awareness for the ks-activity. In this approach, knowledge awareness centers on the knowledge, and benefit from its conceptualization to give appropriated awareness.

5.2.2 Discussion Awareness

Knowledge awareness, as it was presented in the previous section, has a general view about the ks-activity. However, an exhaustive analysis about what knowledge awareness means was not done when we focus the activity on the development of a discussion thread.

In a nutshell, discussion awareness is in charge of making knowledge divergence acceptable. People need to make the distinction whether a new contribution is a reaction, otherwise, they cannot realize of the discussion activity. The activity on a ks-community consists of sending contributions to the shared knowledge workspace. While in real communities people are not only aware of the occurrence of a new contribution, but also they put this contribution into a discussion context by any clue, like a body expression, voice variation, or other; in a digital knowledge sharing environment, people need assistance to maintain the discussion context, for example for interpreting whether a contribution is or not a reaction. There are some cases where identifying whether a contribution is by reaction is easy, because it is explicitly expressed, for example the user contributes with an alternative to an existing k-artifact. However, there are other cases where determining if a contribution is or not a reaction it is not evident, for example when a user contribute with an augmentative k-artifact, because it can be confused as an isolated contribution, as it occurs when ontologies are used. Catching up with the discussion context is a way of being aware of the discussion activity.

In the same way knowledge awareness is the needed awareness to follow knowledge evolution; I understand discussion awareness as the required awareness to follow discussion. Discussion awareness is the part of the knowledge awareness, which is specialized in the knowledge divergence occurrences in the ks-activity, because it provides the information that allows individuals to track the discussion activity performed by other participants over the time . Therefore, *discussion awareness is the needed awareness information to keep a ks-community up-to-date about the discussion evolution.*

Discussion awareness is the means by which individuals track discussion activity because:

- it allows a better understanding of discussion subject, because it identifies and provides information about new discussion contributions in the context of a discussion thread;
- it induces participants' curiosity into the discussion; to wake up the curiosity of users is essential to provoke the discussion activity. It should not affect only users participating in the discussion, but also other users that may be interested in those places full of activities;
- it promotes discussion development. Curiosity works as a stimulus to continue with the discussion by contributing with more complementary or divergent knowledge; and finally,
- it helps discussion evolution understanding; it keeps members up-to-date about discussion progresses through the ks-activity.

Discussion awareness helps people to catch up with knowledge discussion; it gives to the individual information that allows them to answer questions like: *is there new discussion activity? who is participating in the discussion? how has this discussion thread evolved?*

Lastly, discussion awareness makes evident the knowledge divergence occurrence. I have already said that one way to guarantee people's interaction in a CKS system is by means of allowing conflict occurrence, but also that awareness is another good approach to promote collaborative interaction. Therefore, discussion awareness, that is awareness about the discussion activity, or more particularly, awareness of the divergence, is extremely appropriated to satisfy the awareness needs of a ks-community, because it reinforces the advantage that the support of divergence occurrence provides to improve the usability of the CKS system.

Operatively, discussion awareness is the part of the knowledge awareness in charge of identifying highly-active knowledge artifact. A discussion awareness service has to gather information about the new discussion contribution which was performed at the PuW.

5.3 Information Needs

To design a knowledge awareness framework is important to understand what information is necessary. This is information to understand ks-activity at the PuW. This knowledge about the ks-activity identifies, categorizes and explains what information should be tracked and captured when ks-activity occurs and how this information may be useful to the user. A designer has to be concerned with what information elements are relevant to the users, how this information should be gathered from the PuW, how it should be delivered to the user and visualized in the PrW-PuW workspace.

Knowledge awareness tracks all the actions related to ks-activity, like contributions and consuming actions. Although the main focus is put on the publishing actions, since they are the responsible for knowledge evolution; consuming actions are also interesting to track because they may give information about the other participants' interest .

According to the proposed workspace to support the ks-activity, knowledge awareness must take into account that:

- People are developing a shared-KR collaboratively, that is, a conceptualization of the common understanding.
- People discuss their conceptualizations. Divergent contributions are a fundamental component of the ks-activity.
- People work at the two workspaces simultaneously. Knowledge artefact are externalized in the private workspace, while they are published as contributions in the shared workspace.
- People need to be up-to-date of "what is going on with the knowledge" at the shared workspace and articulate this with their individual perspectives.

- People work at different times (asynchronous work), so people must be kept up-to-date on what has happened in the interim. According to the ks-activity, mostly, knowledge awareness is information about the past. Mostly, knowledge awareness is historical information of what happened in the shared workspace. There is no real-time interaction among participants; they only interact by sending contributions.

In order to develop a framework for knowledge and discussion awareness, I have taken Tam's approach [Tam04] to understand the awareness information needs of the community members. I analyze the possible questions that may be asked by the users and the different perspectives they apply to make these questions. To catch up with knowledge activity, individuals need information that allows them to answer this question: *what has happened with the knowledge?*; and the knowledge awareness framework must provide an answer to this question. However, people need track ks-activity from different perspectives. These perspectives are: the *k-artifact perspective*, the *actor's perspective* and *KR perspective*.

- People with a k-artifact perspective are interested in tracking what is going on around a knowledge item, and need answers to questions like *what has been done to this k-artifact? how has this k-artifact evolved?* Knowledge artifact perspective identifies users' knowledge interest. These interests can be in any knowledge item or in those by his/her own authorship.
- People with an actor's perspective are interested in following ks-activities of other members, so they need to answer questions like *who has made this contribution? where has this person been contributing?*
- However, a KR perspective helps individuals to understand where the ks-action is located in the KR. It gives a more general view of the KR as opposed to the k-artifact perspective that focuses on a knowledge item. People with a KR perspective focus in understanding what activities are been carried out. They are interested in questions like *what contributions and other events have occurred in the KR?* For example someone can be interested only in publishing or consuming actions or in discussion actions.

Each perspective state high-level questions that people then turn into more specific questions. These refinements allow them to get more precise information about the ks-activity. These tentative questions are:

- *Where* has the ks-activity taken place?
- *Who* has performed the ks-activity?
- *What* (which) k-action has been performed?
- *How* was the ks-activity performed?
- *When* did the ks-activity take place?
- *Why* was the ks-activity performed?

These questions help to understand which knowledge is necessary to get a suitable understanding about what is going on with the community's knowledge. These perspectives in combination with the six previous question categories give us a framework to analyze the knowledge awareness information needs.

This analysis applies to both knowledge and discussion awareness. However, when discussion awareness is considered, there are additional specific questions and perspectives to take into account. In discussion awareness, the interest is in the discussion evolution, and then questions like *is there any discussion?* should be answered. Because of this duality, I suggest a separated analysis about the information needs; one from the general point of view of the ks-activity (information needs of the knowledge awareness 5.3.1), and the other paying special attention to the discussion activity (information needs of the discussion awareness 5.3.2).

This knowledge awareness framework is developed independently from the knowledge representation system. It might be applied to any virtual environment that supports a collaborative development of the KR which presents k-artefact as its building blocks.

5.3.1 Information Needs of Knowledge Awareness

In this section, I will present the knowledge awareness information needs through the analysis of where, who, what, how, when and why questions and by taking into account the k-artifact, actor's and KR perspectives. As result, it is possible to identify information elements. This analysis facilitates the identification of what information should be tracked by a knowledge awareness mechanism. Some of them are information about location, gazing, presence, readership, contributor, actions, process, and motivation. Many of these *information elements* were taken from Gutwin's workspace awareness framework [Gutwin02], and adapted to the knowledge awareness requirements. To present details about each question category and perspective I will present each question and analyzing their impact from the different perspectives.

The "Where" question

The "where" question (where-question) indicates where the action occurs from a physical point of view. This question needs a space to locate the ks-activity. As the ks-activity is carried out at PuW, which hosts the KR, it is the KR the spatial reference to locate the ks-activity. The "where" question is strongly related to the structural organization of the KR. For example in an ontological approach, the KR is structurally organized at a conceptual level and at a concrete level; and the conceptual level could be organized in classes, relationships, and so on. Even, ontological conflicts are represented as locations in the KR (we may also identify the location of conflicts in the KR).

Where-questions provide information about the location of ks-activity; this activity mainly locates publishing and consuming actions. These activities will be presented from the three perspective answering more specific questions like: where was a k-artifact in the KR?, where was an individual participating in the KR? and where has ks-activity taken place at the KR?

Table 5.1.: The Where question

where			
Information elements	Perspectives		
	Knowledge artifact (ka)	Actor	Knowledge Repository (kr)
Location	where was a <i>ka</i> when I left?	where has somebody visit the <i>kr</i> ?	where (exactly) have people been in the <i>kr</i> ?
Gaze	where is it now?	where (exactly) in the <i>kr</i> has a somebody been looked at?	where was the <i>ka</i> located within the <i>kr</i> ?
Contribution	where was this <i>ka</i> while I was away?	where (exactly) in the <i>kr</i> has a person directed the contributions?	which part of the <i>kr</i> have people looked at? which part of the <i>kr</i> have people made contributions to?

Where-questions identify location, gaze, and contribution as the information element, which are needed to offer a suitable knowledge awareness. The next table 5.1. represents the where-question. *Location information element* gives historical information about where k-artefact and actors were located into the KR during the ks-activity. Location makes reference to more general activities that take place at the KR as knowledge retrieving activities. Although, at the beginning, the k-artifact's location is not quite relevant, but it will gain importance when conflicts are considered because this allows locating conflicts at the KR.

On the other hand, *gaze information element* gives historical information about more precise activities in the KR, such as consuming actions. Although, both location and gaze information element show a passive activity as regards the knowledge evolution, they serve to locate potential places where future contributions may take place, and they are even able to understand other participant interests. Gaze and location information are also useful to articulate shared knowledge with private ones.

Finally, the *contribution information element* refers to the most critical activity to locate. Contribution information identifies places in the KR where the publishing action is being or has been carried out. It is really relevant because it locates where the action is within the KR. It also locates high-level activity, li discussions.

Where-information can be about the present or past of the activity. Although most of the information is about the past of the activity, information about the present location of the activity is important to give clues about when it is needed to synchronize the shared and individual-KR (in real-time).

The "Who" question

In ks-activity, knowing who has made a contribution represent an opportunity to stimulate the ks-activity, and in particular, the knowledge interaction. On many occasions, people are interested in knowing about the other members' activities. People can pay attention to knowledge contribution depending on who has made this contribution. And depending on who has made a contribution, people may or not react (whatever this reaction means: augmentative or divergent).

Mainly, who-question identifies the individual who has performed a ks-actions at the KR. These activities are analyzed from the three perspectives, so that, by answering questions like: who has made contributions? who made this contributions? who works with whom? and others.

Table 5.2. shows the appropriated questions to the who-category. These questions give different information elements such as: presence, identity, readership, and contributor. *Presence information element* gives information about who is or has been at the PuW. Presence it is not so relevant because this approach does not focus on real-time interactions among users.

However, *identity information element* allows identifying relationships of proximity among individuals, and consequently identifying "indirect" interactions. For example, identity is useful to be able to answer questions like: who is participating close to me? (two users are participating closer if they are more active around the same k-artefact). Therefore, this category promotes a better understanding of what the reasons that have moved a user to perform certain action were.

Readership information element helps to know who retrieves knowledge from the KR. It allows understanding if other members have agreed or not with his/her own contributions; for example, if a user may know that other members are visiting her/his contributions and then the visitors do not react, s/he may assume that the visitors have agreed with her/his contributions.

Finally, the *contributor information element* identifies who makes publications at the KR; who has published k-artefact. The contributor is the more interesting element of the who category, because it actually allows identifying who is "active" in the ks-activity.

The "What" question

What-question allows tracking the activity's history inside the KR. Activity history models low-level actions that were performed in the KR. These low-level actions are those which are supported by consuming and publishing in the KR. Therefore, what-question gives knowledge about the ks-actions which were performed in the KR. For people it is easier to understand what has happened in the KR in terms of the performed low-level actions; because, people are used to talking about the past in terms of the occurred actions. The What-question is the easiest question to detect by designers. It is because this actions are supported on the KR system.

As I have said above, the what-question tracks ks-actions. These actions are always those for consuming and publishing. Studying these activities from the three perspectives provides, as result, questions like: what contributions have taken place? and what actions

Table 5.2.: The Who question

who			
Information elements	Perspectives		
	Knowledge artifact (ka)	Actor	Knowledge Repository (kr)
Presence (P)	Who has browsed this <i>ka</i> ?	Who has this person "interacted" with?	Who has been in the <i>kr</i> ?
Identity)	Who has queried this <i>ka</i> ?	Who made contributions with this person	Who has looked at the <i>kr</i> ?
Readership (R) Contributor	Who has contributed to this <i>ka</i> ?		Who has made contributions to the <i>kr</i> ?

has a user done?

Table 5.3. shows in details the what-questions. It only identifies the *action information element*. Knowledge artifact perspective of an action simply answers the question: "what contributions have been made to this k-artifact?". This question implies to take into account the context of the k-artifact. Depending on the knowledge representation system, this question can be considered at different level of granularity. For example in an ontological approach, the addition of a new slot to an existing class (the k-artifact) could be an augmentative contribution to the class (a low-level action) and may be interpreted as a high-level action - class *updating*. Low-level action can be put in the context of high-level goals as it was shown in the previous paragraph, but in this way a high-level action history is more related to the intention of the action, being a characteristic of the why-question. From the perspective of an actor, the what-question shows what actions an actor has executed over a particular knowledge artifact. Although, every action can be tracked, contribution actions are those giving more knowledge about the knowledge evolution. Lastly, the KR perspective contributes with knowledge about the actions that were performed in the KR, for example what actions have been performed at the KR? or what k-artefact have experimented any action?

The "How" question

The "how" question asks how the current KR differs from how it was before. It identifies the process carried out to achieve the current state of the KR. And thus, people can get an idea of evolution of the knowledge. To have only knowledge about instantaneous contributions it is not enough to get a complete understanding about the knowledge evolution, because it may require an additional cognitive effort on behalf of the user to get a suitable idea about what was going on.

Therefore, how-question put ks-activity in the context of the knowledge evolution; therefore, it is useful to understand the operational details that took the KR from a previous state to the current one.

Table 5.3.: The What question

what			
Information elements	Perspectives		
	Knowledge artifact (ka)	Actor	Knowledge Repository (kr)
action	What contributions have been made to this <i>ka</i> ?	Which <i>ka</i> has somebody seen? Which <i>ka</i> has somebody contributed to? Which activities has somebody engaged in?	What contributions have taken place at the <i>kr</i> ? What queries have occurred at the <i>kr</i> ? Which browsing has taken place?

Interpreting how-question from the different three perspectives allows understanding how a k-artifact or the KR has evolved, and how the actors participate in their evolution.

How category can reflect the how-knowledge as the process history or as the outcome history. Processes make an abstraction about the action history in order to find out high-level action and explain ks-activity in term of ks-process steps. For example, it interprets a contribution as a reaction to another contribution. The most relevant process abstraction is the discussion process, which will be discussed in section 5.3.2. On the other hand outcome point of view just refers to highlighting the difference between two KR versions, the original version and the current one. Between both approaches, a process's history is more suitable to follow knowledge evolution, while outcome approach is more appropriate to focus on knowledge changes.

In Table 5.4., I have represented the knowledge element corresponding to the "how" category. A user with a k-artifact perspective needs information about how a particular k-artifact has evolved. On the other hand, a user with an actor perspective needs knowledge about how other users have shared knowledge, and thus, he/she may know how things have evolved. In terms of the KR perspective, information about the evolution of the KR is required.

The When question

Knowing the sequence of the occurred events is essential to achieve a suitable notion of knowledge evolution. The when-category deals with this feature giving a sequential order to these events. When-question give a chronological context to the other categories. For example, if someone browses a knowledge artifact and then makes an augmentative contribution, it is possible to deduce that this individual is interested and agrees with the former contribution, because he/she complements it. In this example, the sequential order is essential to deduce "why information", why has this individual performed this contribution? – the intention.

In Table 5.5. the different perspectives of the question *when has the ks-activity oc-*

Table 5.4.: The How question

how			
Information elements	Perspectives		
	Knowledge artifact (ka)	Actor	Knowledge Repository (kr)
Process	How has this <i>ka</i> evolve?	How has this person made contributions?	How has the <i>kr</i> evolved?
Outcome		How has this person shared knowledge? How has this person made things evolved?	

curred? are shown. Events are the information elements in this category. The table mainly focuses on those events that really imply knowledge exchange, as it is contribution events. Chronological tracking of the other actions may also be useful to understand the user's activity.

The Why question

The "why" category is useful to understand the intentions and reasons of the contributions. It helps the user to get an idea about the cognitive and motivational causes to perform a ks-action. According to Tam and Greenberg [Tam04], cognitive history describes the logic or reasoning that may be behind a ks-action, which is a rational reconstruction of the person's goals and plans. Motivational history deals more with the impulses or desires that are the impetus for making a ks-action, which is the actual reason why a person did something. The causes of why they are separated elements is because a contribution may be based upon a well thought out and carefully conceived plan or it may be a more impulsive reaction to the current situation. On the other hand, to capture both kinds of information is not really possible to be done automatically. Computers are not so intelligent to deduce cognitive or motivational intentions. However, in a knowledge sharing workspace like this, where ks-action has an explicit overloaded representation of the meaning of the intention (i.e. augmentative or conflictive contribution) it is possible to gain a degree of automatization of the intention or motivation of a performed action.

Why category should answer questions like: why was this contribution done? As the other categories it can be analyzed from the three perspectives: k-artifact, actors and group memory. Depending on the different perspectives, questions like those presented in Table 5.6. can be posted with the users.

Besides, why category is an essential component to help to understand the discussion thread. Because the question *why was a contribution made?* may be interpreted as *is this contribution a reaction to a previous contribution?* It is a clear example of a cognitive cause of the contribution, because it explains the goals and plans of the use. The "why"

Table 5.5.: The When question

when			
Information elements	Perspectives		
	Knowledge artifact (ka)	Actor	Knowledge Repository (kr)
event	When was this <i>ka</i> published?	When did a person make contributions?	When were contributions to the kr made?
	When was this <i>ka</i> consumed?	When did a person contribute with a particular ka?	When did this particular contribution occur in the kr?
	In what order were contributions made to this <i>ka</i> ?	In what order did a person make contributions?	In what order were contributions made to the kr?
		When did a person consume knowledge?	When was the knowledge consumed?

category becomes a fundamental component of knowledge awareness when there is no specific awareness which supports the discussion activity. This discussion will be delayed to next section.

5.3.2 Information Needs of Discussion Awareness

Previous section has presented the information needs that knowledge awareness covers by taking a general approach of the ks-activity. However this section focuses on the discussion activity. As was previously defined in section 5.2.2, discussion awareness is information to keep a knowledge-sharing community up-to-date about the discussion evolution, that is, to be aware of the discussion activity. Although discussion activity is part of the ks-activity, in this section it is analyzed in an isolated way, where it only focuses on discussion actions.

To catch up with discussion activity, individuals need information that allows them to answer questions like this: *Are there new discussion occurrences?* It is responsibility of the discussion awareness framework to give answer to these questions.

Similarly knowledge awareness, discussion awareness should give enough information to be able to answer next high-level questions. These questions aid to comprehend what knowledge is necessary to have a suitable understanding about what is going on with a discussion, and consequently, understanding what is going on with the knowledge.

- *Where* has discussion activity taken place?
- *Who* has performed the discussion activity?
- *What* (which) discussion activity was performed?

Table 5.6.: The Why question

why			
Information elements	Perspectives		
	Knowledge artifact (ka)	Actor	Knowledge Repository (kr)
Cognitive	Why was this <i>ka</i> published?	Why did this person make contributions with this <i>ka</i> ?	Why was this contribution made to the kr?
Motivational			Why was this browsing made to the kr? Why was this query made to the kr?

- *How* has the discussion activity been performed?
- *When* did discussion activity take place?
- *Why* has discussion activity taken place?

Giving an answer to the previous questions allows tracking the discussion activity, focusing on the discussion contribution actions (since they are the responsible for making evident the divergence occurrence). Discussion actions such as augmentative and conflictive contributions and argumentations are the actions to be monitored; and the discussion artifact are the k-artefact attached to these actions. In case of the consuming actions it is possible to apply the general approach of knowledge awareness but in the context of the discussion thread. In the reminder of this section, consuming action will be mentioned only when a special treatment is required in the context of discussion awareness; otherwise the knowledge awareness approach should be applied.

Once a discussion has been triggered through an open discussion action, the discussion awareness framework tracks the activity around the discussion thread. Therefore, the discussion thread becomes the place where to locate most of the discussion activity.

People need track discussion activity from different perspectives. These perspectives are almost similar to those defined to the knowledge awareness: *discussion artifact perspective*, *actor perspective*, *discussion thread* and *KR perspective*.

- People with a k-artifact perspective are interested in tracking what is going on around a discussion artifact, and need answers to questions like *what has been done to this discussion artifact? how does this discussion artifact evolve?* From a discussion point of view, the discussion artefact do not only represent the k-artefact, but also they mainly represent a discussion component (alternative, divergence, argument, etc).

- People with an actor's perspective are interested in keeping up the participation of other members on a discussion, so they need to answer questions like *who has made this discussion? where has this person been discussing?*
- People with a discussion thread perspective center their attention on the activity at the discussion thread. Physically, this perspective focuses on the discussion thread development. This perspective gives a detailed view of the discussion thread evolution.
- People with a KR perspective focus on understanding what discussions are being carried out in the KR. This perspective only concentrates on discussion actions. In this perspective, people can be interested in knowing what discussion was open, where, why, etc. This perspective is the same as in knowledge awareness, except that here it focuses on the discussion activity. This perspective gives a general view of discussions at the KR.

These four perspectives in combination with the six previous question categories give us a framework to analyze the discussion awareness information needs. In the remainder of this section I will present the discussion awareness information needs through the analysis of the where, who, what, how, when and why questions, taking into account the k-artifact, actor and KR perspectives. As result, it is possible to identify information elements. This information facilitates the identification of what information should be tracked by the discussion awareness mechanism. Some of them are information about location, presence, contributor, actions, process, and motivation. To present details about each question category and perspective, the style used in section 5.3.1 is followed, where each question is presented and then analyzed from the different perspectives.

The Where question

The "where" question indicates where the discussion takes place from a physical point of view. In this category the physical space corresponds to the discussion thread, except the action to open a discussion which takes place at the KR space. The where-category is strongly related to the structure of the discussion thread. It corresponds to the same structure described in section 3.5.1, where there is an initial contribution, from which other discussion contributions (augmentative or divergent discussion artefact and argumentations) are linked.

Where-question gives information about the location of the discussion activity; it mainly locates where discussion actions take place. This category will be presented from the four perspectives which answer more specific questions like: where was a discussion artifact in the discussion thread?, where was an individual participating in the discussion? where have discussion actions taken places at the KR? and others. While the KR perspective allows locating the discussion places, the discussion thread perspective allows locating the discussion contribution inside the thread.

Table 5.7. represents the where category. The discussion information elements are: *location*, *gaze* and *discussion contribution*. Location gives historical information about where discussion artefact and actors were located in the discussion thread during the

Table 5.7.: The Where question for discussion awareness

where				
Information elements	Perspectives			
	Discussion artifact	Discussion actor	Discussion Thread	Knowledge Repository
Location	where was a <i>da</i> when I left	where in the <i>dt</i> has a person contributed?	where has an <i>discussion contribution</i> been made at the <i>dt</i> ?	where was a discussion in the <i>kr</i> opened?
Discussion Contribution	where is <i>it</i> now? where has this <i>da</i> been while the time I was away?	Which discussion has a person been engaged in?		

discussion activity. At the beginning, the discussion artifact's location is not quite relevant, but it becomes important when they are considered as conflicts locations in the KR. Thus, the discussion thread is considered as a "knowledge artifact". Location makes also reference to less relevant actions that take place at the KR such as consuming actions. Gaze information elements are the same as in knowledge awareness, except that here they work on the top of the discussion thread.

Among all information elements, discussion contribution is the most relevant one. It places the discussion activity into the discussion thread and even into the KR; consequently it identifies places where the discussion activity has been carried out.

The Who question

The who-category is useful to know who has carried out discussion activity. It gives information about the others' discussion activities. People can pay attention to discussion contributions depending on who has made the contribution. In a discussion activity, the who-category plays an essential role in motivating discussion, more than in ks-activity. People involved in a discussion activity are more reactive than people who only share knowledge. Many times, an individual that makes a discussion contribution does not only wait for reaction contributions, but s/he also waits for specific members' reactions.

Who-question identifies the individual who has performed a discussion activity, as such to open a discussion or to make a discussion contribution. These actions are analyzed from the four perspectives so that answering questions like: who made discussion contributions? who participates in this discussion? who discusses with whom? who opens the discussions? and others.

Table 5.8. shows the appropriated questions to the who-category. These questions give different knowledge elements as: presence, identity, readership, and contributor.

Table 5.8.: The Who question for discussion awareness

who				
Information elements	Perspectives			
	Discussion artifact	Discussion actor	Discussion Thread	Knowledge Repository
Presence	Who has contributed with a <i>da</i> ?	Who has this person "discussed" with?	Who has made a discussion contribution to this <i>dt</i> ?	Who has opened discussions?
Identity	Who has contributed with this <i>da</i> ?			Who is discussing in the KR?
Contribution				

Presence information element gives information about who is or was participating at a discussion thread; however as the collaborative activity is not necessarily a synchronous activity, people need presence information of the past, and it can be deduced from the contributor or identity categories.

However, identity information element allows identifying proximity relationships among individuals, and thus, identifying discussion interactions among other users. For example, identity is useful to answer questions like: who is discussing with me? and in consequence to understand better the reason of some replays.

Contributor information elements identifies who makes contributions to the discussion. Contributors are the most interested element of the who-category, because it really allows identifying who was participating (being active) in the discussion. Contributors are those who have published some discussion artifact to the discussion thread or have opened a discussion.

Readership information element is like in knowledge awareness, except that here it helps one if other members agree or not with his/her discussion contributions. For example, if a user sees that another member is visiting her/his discussion contribution and then the visitors do not react; then s/he can assume that the visitors agree with their discussion participation. Readership category is not shown in the table 5.8..

The What question

The what-question allows tracking the history of a discussion. Discussion history models low-level actions that were performed in the context of a discussion. These low-level actions are mainly those corresponding to discussion contributions or open a discussion. People are used to following the dynamics of a discussion in terms of giving complementary information or different points of views. Therefore, working in a discussion-supported environment that supports these activities as low-level action becomes a natural practice for discussing. Besides, what-question is the easiest question to detect by designers, since these actions are underlying in the system.

Table 5.9.: The What question for discussion awareness

what				
Information elements	Perspectives			
	Discussion artifact	Discussion actor	Discussion Thread	Knowledge Repository
action	What discussion contributions have been made to this <i>da</i> ?	What discussions has a person opened? Which <i>da</i> has a person contributed?	What discussion contributions have taken place at this <i>dt</i> ?	What discussions were opened?

Low-level discussion actions are those for making discussion contributions. Analyzing these actions from the four perspectives gives the following questions: what augmentative contributions have taken place? and what discussion actions has a user done? However in the context of the KR perspective, the what-question identifies the opened discussion.

Table 5.9. shows in details the what-question. Discussion artifact perspective of actions, simply makes the question: *what discussion contributions were attached to a discussion artifact?* This question involves taking into account the context of the discussion artifact; it may be a discussion thread or one of its components. For example, the addition of a new augmentative contribution or an argument are actions applied to a discussion thread component, while the action open a discussion thread "create" a discussion thread in the KR. From the perspective of the actor, the questions show what discussion actions the actor has performed. Lastly, the discussion thread perspective contributes with knowledge about the actions that were executed in a discussion thread, but they may also be considered in the KR; for example what discussions were opened at the KR?

The How question

The how-question asks how a discussion thread differs from how it was before. It identifies the process carried out to achieve to the current state of the discussion thread. Therefore, people can get an idea of the evolution of the discussion. To have only knowledge about instantaneous contributions it is not enough to get a complete understanding about the discussion evolution.

Therefore, how-question puts discussion activity in the context of the discussion evolution and thus, in the context of the knowledge evolution. Therefore, it is useful to understand the operational details which the discussion thread took from the previous state to the current one.

Interpreting how-question from the different perspectives allows understanding how a discussion artifact or a discussion thread has evolved, and how the actors participate in their evolution. These questions give an abstract point of view of how the KR has

Table 5.10.: The How question for discussion awareness

how				
Information elements	Perspectives			
	Discussion artifact	Discussion actor	Discussion Thread	Knowledge Repository
Process	How has this <i>da</i> been discussed?	How has this person discussed?	How have augmentative/conflictive contributions or argumentations made the <i>dt</i> evolution?	How have discussions evolved in the KR?
Outcome	How has this <i>da</i> evolved?	How has this person made discussions evolve?	How has the <i>dt</i> evolution taken place?	

evolved.

In Table 5.10., I have represented the information elements corresponding to the how category. They are process and outcome information elements. A user with a discussion artifact perspective needs information about how a particular k-artifact has been discussed; but the answer to this question is the discussion thread. On the other hand, an actor's perspective involves knowledge about how a user has discussed, and how s/he made discussion evolution. In terms of the discussion thread perspective, information about how discussion actions have influence on the discussion thread evolution is required.

The Why question

The why category is useful to understand the intentions and motivation that moves people to be involved in a discussion. It helps users to get an idea about the cognitive and motivational causes of performing a discussion activity. Why category should answer question like: why has this occurred?

As the other categories, it can be analyzed from the three perspectives: knowledge element, actors and discussion thread. Depending on the different perspectives, questions as those presented in Table 5.11. can be posted with the users.

The why-category it an essential component to help to understand the discussion thread. Because the question *why was a contribution made?* may be interpreted as *is this contribution a reaction to a previous contribution?* It is a clear example of a cognitive cause of the contribution, because it explains the goals and plans of the user.

The When question

Knowing the sequence of the occurred events is essential to have a suitable notion of the discussion evolution. The when-category deals with this feature giving a sequential

Table 5.11.: The Why question for discussion awareness

why				
Information elements	Perspectives			
	Discussion artifact	Discussion actor	Discussion Thread	Knowledge Repository
Cognitive	Why was this <i>da</i> published?	Why has this person opened a discussion?	Why was this discussion contribution made to the <i>dt</i> ?	Why were discussions opened at the KR?
Motivational		Why did this person make this discussion contribution?	Why was this <i>dt</i> opened?	

order to the discussion events. When-question give a chronological context to the other categories.

In Table 5.12. the different perspectives of this question are shown: when a discussion activity was carried out. As in the other categories it is presented in terms of events (by means of information elements and perspectives). The table mainly focuses on those events which really involve knowledge exchange, as discussion contribution events are.

5.4 Knowledge Awareness Mechanism

The knowledge awareness mechanism is a software component of the CKS system that is responsible for keeping the community up-to-date about the ks-activity. As the most of awareness mechanism, knowledge awareness mechanism focuses on two main functionalities: the gathering and delivering of awareness information. Gathering awareness information involves the collection of information about the collaborative activity, while the delivering awareness information involves the dissemination of gathered information to the users. Therefore the knowledge awareness mechanism is made up of two components: one to gather awareness information and other to deliver awareness information.

Gathering knowledge awareness information involves an implicit knowledge collection about the ks-activity. This knowledge collection involves a component that observes the community to collect knowledge about members' activities in the shared knowledge workspace (awareness gathering component). This component reacts when an observable event occurs in the context of the PuW. Observable events are those events that involve some activity on the top of knowledge. In the context of the PuW these events are those derived from ks-actions. Gathering component has to be able to collect the necessary information to satisfy the information needs presented in section 5.3.

The gathering component is also concerned with the activity at the PrW, because some change at the private version can involve a "divergence" with the shared version, but this

Table 5.12.: The When question for discussion awareness

when				
Information elements	Perspectives			
	Discussion artifact	Discussion actor	Discussion Thread	Knowledge Repository
even	When was this <i>da</i> contributed?	When did a person make discussion contributions?	When was this <i>dt</i> opened?	When were discussions opened?
	When was this <i>ka</i> discussed?	When did a person contribute with a particular <i>da</i> ? In what order did a person make discussion contributions? When did a person open a discussion?	When were discussion contributions to this <i>dt</i> made? In what chronological order were discussion contributions made to the <i>dt</i> ?	In what chronological order were discussion opened in the <i>dt</i> ?

change do not affect the community because it remain hidden in the PrW. However, individuals need some mechanisms that keep them aware about her/his private version remains divergent with the shared one. This mechanism is rather simple, because it only to have to pay attention of the shared k-artifact and "mark" them when they are changes at the PrW; while the gathering of changes of the shared k-artefact at the PuW is covered by the gathering of knowledge awareness information component which has been describe above.

On the other hand, delivering knowledge awareness information involves a component that disseminates awareness information. As it was previously mentioned, the awareness information is deduced by the gathering component; therefore, the main responsibility of the delivering component is to guarantee that this information arrives at user according to her/his needs. The delivering component should be able to adapt delivered awareness information according to individual knowledge contexts. This means that the delivered knowledge awareness information should be accorded to the user's profile: interest, expertise, social relationships and the activities where they are engaged. Although, personal casting, broadcasting, and subscription are typical mechanisms to know the user's profile and be able to distribute suitable knowledge awareness; a more dynamic user's model, it is needed to represent the activities in which users are engaged.

Both, gathering and delivering components need to work in collaboration with member's profile. The member's profile is a user's model, which is partially represented in the KR by means of the members' expertise and interest, but it may also model the user's

activity.

In summary, knowledge awareness mechanism works as follows: first, it captures the low-level information of the ks-activity, then, it generates high-level information about the ks-activity and finally, it delivers knowledge awareness information to the users.

In the next sections (5.4.1 and 5.4.2) I will present details of the gathering and delivering components to implement a knowledge awareness mechanism, and as a side effect, I will also introduce the member's profile component.

5.4.1 Gathering of Knowledge Awareness Information

Based on the observable actions that occur in the PuW, the gathering component has to be able to collect and process the necessary information to answer the question presented in section 5.3. The gathering component involves both, the implicit collection of knowledge about the ks-activity and the mining of high-level information about ks-activity. The implicit low-level knowledge collection involves having a component that observes the ks-activity in order to capture low-level information about the ks-actions which were performed at the PuW (what was the performed action, who has performed it, and what knowledge is involved). On the other hand, to determine high-level information about the ks-activity, this component mines the low-level information in order to discover more useful information, which is the high-level information.

Capturing Low-level Information

In the context of the PuW, the ks-action to take into account are not only those derived from contribution activities but also from consuming actions. Most of these actions are made up of some observable events, for example a browsing action involves many link selection events. A link selection event only involves opening, by a click, the interface of a knowledge element. However, a browsing action is a sequence of link selection events. A querying action involves an event to open the query editor, another one to write the query, another one to execute the query and another one to inspect the result. On the other hand, contributing actions involve only one event, because the only event that occurs at the PuW is the act of publishing. While events are low-level ks-activity units, actions are high-level ks-activity units, except for publishing actions that will be considered as a low-level event in the context of the ks-activity, and discussion actions that are low-level events in the context of the discussion thread.

In this thesis, I will focus on only on those activities related to publishing and discussion actions. Because consuming actions which are considered as abstractions of low-level events, need another approach be deduced. However, it is possible to face up consuming actions by following the Christoff Bouthier's approach [Bouthier04] where high-level information is deduced by means of understanding the context where the low-level events have occurred. His approach is based on *Bayesian Networks* to classify the occurred event as part of a consuming action.

Therefore, the gathering component has to consider publishing and discussion actions as low-level events, and then to put them in the context of a ks-activity. When an event occurs, information about what the event is, who performs it and , what knowledge

artifact is involved should be captured. It is responsibility of the PuW to capture this information. This event information is captured directly from the PuW and it is low-level information. This information consists of the ks-action, its performer (the user) and the involved k-artifact, as it is shown in Figure 5.2.

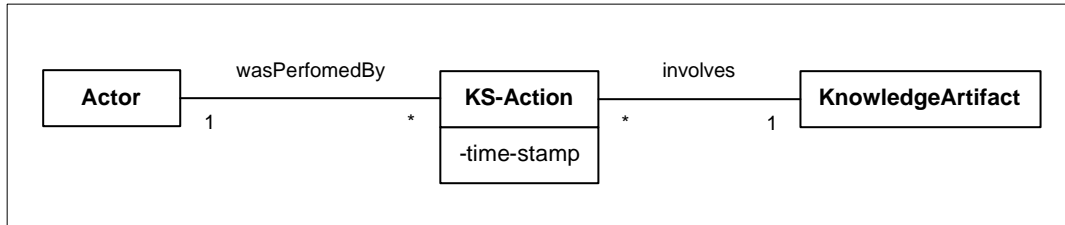


Figure 5.2: The conceptual model of a low-level knowledge-awareness information item

With this minimal amount of information it is possible to give an answer to the low-level information needs. Next, I show a minimal analysis where I present which low-level questions are possible to answer:

Where-category It locates the ks-action at the KR. As it has been already said in section 5.3, the simplest location at the KR is a k-artifact. The identification of the knowledge artifact covers the two perspectives: actor and KR perspectives. Because

- it is the the k-artifact where the actor has performed a ks-action, and
- it is the k-artifact where a ks-action has occurred at the KR

Who-category It identifies the actor, that is the performer of the ks-action. This covers the two: k-artifact and KR perspectives:

- it is the actor who has performed a ks-action over the k-artifact,
- it is the actor who has performed a ks-action at the KR

In this category, there is no actor perspective because actors perform ks-actions in isolation.

What-category It determines the performed ks-action. It gives low-level information to the three perspectives, because:

- it is the ks-action that has occurred over the k-artifact,
- it is the ks-action that has been performed by an actor
- it is the ks-action that has occurred in the KR

When-category It identifies the time-stamp of a ks-action. It is the instant when a ks-action has been performed by an actor in the KR to manipulate a k-artifact.

How-category It is the ks-action in the context of the ks-process. It identifies the kind of the ks-action (for example, it determines if the ks-action is a publishing action or a discussion action). It gives low-level information to the three perspectives, because:

- it identifies what kind of ks-action has occurred over the k-artifact,
- it identifies what kind of ks-action has been performed by an actor
- it identifies what kind of ks-action has occurred at the KR

Why-category It determines the intention of the ks-action. It is given by the kind of the ks-action. It gives low-level information to the three perspectives, for example in case of a publishing action:

- the intention is to contribute with a k-artifact,
- the intention of the actor is to make a contribution,
- the intention is to contribute to the KR

And in case of a discussion action, for example an opening discussion action:

- the intention is to discuss around a k-artifact,
- the intention of the actor is to open a discussion,
- the intention is to open a discussion to the KR

Providing a information items regarding a structure as is shown in Figure 5.2, is the simplest approach to provide low-level knowledge-awareness information or, in shot low-level notifications. This approach can be also applied to discussion activity with minor changes. These changes involves focusing the k-artifact perspective on discussion artifact and adding a new perspective: the discussion thread perspective. The discussion thread perspective places the discussion activity in the discussion thread and consider it as a k-artifact of the KR.

From low-level to high-level knowledge-awareness information

Low-level information gives a partial answer to questions of section 5.3. However it is not obvious how it is possible to answer questions like how, why or more sophisticated instances of questions where, when, who, what.

As the main goal of knowledge awareness (including discussion awareness) is to provide awareness information that people need to understand the ks-activity which has been carried out. People need information about what happened at the KR while they have been away. People, mainly, need to know the history of the activity. Histories cover the period of time while they have been away and there are different categories of histories. One for each information need category. Below, I will present each one:

Where-history It temporally locates where the ks-activity has been carried out at the KR. It defines a collection of KR places, which according to the different perspectives are:

- the k-artefact where the actors have performed ks-actions.
- the k-artefact where ks-actions have occurred at the KR

Who-history It temporally identifies the actors' activity.

- who has been performed ks-actions over a k-artifact?. This means the collection of users that have performed a ks-action over a k-artifact.
- who was collaborating with an actor? This means the analysis of the discussion threads to collect pairs of actors.
- who has been active at the KR? This means the collection of users who have been active in the KR.

What-history It sorts chronologically the ks-activity. The history of the activity can be seen from three perspectives such as:

- what ks-actions have occurred over a k-artifact?
- what ks-actions have been performed by an actor?
- what ks-actions have occurred at the KR?

When-history It is covered by the other categories.

The where, who and what histories are subset of the collection of low-level notifications which have been selected in accordance with an certain criterion. They are easily implemented by keeping them in some kind of storage. For example, in the ontological approach presented in Chapter 4, they may be the knowledge-base. In this case, it is possible to use the appropriate query language to obtain the corresponding history. For example, in the prototype application which will be introduced in the next chapter, I have taken this approach.

However, building histories of the other knowledge categories (how and why) becomes less simple. Different approaches may apply to different aspect of the history category. Next, I will only name these history categories in order to notice the kind of knowledge they represent. However, I just suggest some approaches that may be taken to design a computer-supported solution.

How-history Depending on the perspective, it shows different points of view of the ks-activity evolution.

- how has this k-artifact evolved? It is embedded on the discussion thread.
- How has this person shared knowledge? This shows patterns of users' behavior.
- how has the KR evolved? It is necessary to identify "critical" points in the KR. The easiest approach to find out critical points is by means of the identification of the discussion threads.

Some features of how-history are possible of covering by the discussion thread construction, such as those coming from the k-artifact and KR perspectives; however those coming from the actor perspective need to tackle them with more sophisticated techniques. Discovering of patterns of users' behavior may be useful both to predict users' behavior and to understand users' intentions.

Data mining technique may be suitable in this situations. Depending on the information needs, there exist different data mining techniques that may be useful. For example, techniques for discovering associations may help to mine behavioral patterns.

Why-history It explains why a certain ks-actions were performed. It should provide high-level explanations at different level. For example, it would be useful to discover implicit discussion threads that have not been performed by explicit discussion actions, such is the case of a discussion thread of only augmentative contributions. Why-histories are difficult to develop. Because, as I have already mentioned, computers are not so intelligent to automatically deduce the cognitive or motivational intentions. However, some data mining approaches can be useful. For instance, in this situation, it also applies Bouthier's approach to contextualize a ks-action in order to discover its high-level intention. By using the "why channel", it is possible to deduce the role that this ks-action plays in a high-level process. The "why channel" is a bayesian network that allows classifying the ks-action in a high-level intention cluster. This approach will be really useful when we are before a highly active and sufficiently community; otherwise, it is impossible to set an appropriate bayesian network when there is a small input data to mine.

5.4.2 Delivering Knowledge Awareness Information

After gathering knowledge awareness information, this information should be delivered to the users. However, knowledge awareness will be really effective to keep the community "in action" if it helps individuals to internalize knowledge. Keeping people aware of ks-activity it is a way to promote individuals' curiosity and therefore, individual learning. However, the delivered awareness information should be necessary and enough. Too much information may become annoying and then it will produce the contrary effect. Besides, the delivered knowledge awareness information should be tailored to the individual's interest and capability, because different individuals needs different stimuli to react.

A knowledge awareness delivering mechanism should also take into account that individuals alternate between the PrW and the PuW workspaces and they mostly works in an asynchronous fashion. Therefore, individuals need to be notified of the ks-activity, but they also need to be notified whether her/his individual-KR is out-of-date from the shared one.

It is also the responsibility of the delivering mechanism to be in charge of the indirect externalization and its notification. Indirect externalization allows automatically updating the individual-KR with the new knowledge occurrences in the shared-KR. These new knowledge in the shared-KR can be automatically "transferred" to the private one, like transference action, but in this case, it is the system which execute the action, therefore indirect externalization can be see as a knowledge contribution in the other direction. Knowing the individuals, it is necessary to aid them in updating the private knowledge version.

Delivered awareness information is broadcasted to each community members but it should be tailored according to the individual's needs. Interests and expertise features

have to be taken into account to delivery the right amount of knowledge awareness information. Interests define individual concerns about k-artefact. Interests allow one to calculate the appropriate information according to the receiver's needs. Expertise means the level of competence that members have and it determines the quality of knowledge that the receiver needs. Interests and expertise are therefore useful to personalize the delivered knowledge-awareness information so as to be more effective when promoting individual internalization, and as a consequence, the community activity.

Consequently, the knowledge awareness delivering component has to take into account the member's profile. A delivering mechanism should work in cooperation with a member profiles like those already presented in section 4.3.2 (member profile ontology). Thus, the profile can be used as an adapter of knowledge-awareness notifications. Then, any knowledge-awareness notification would be delivered to each member according to her/his profile. Because, in the role of adapter, the member's profile works as filter and allows one to determine which notifications (both low-level and high-level notifications) should be delivered to each user.

In groupware applications of this kind, the responsibility of maintaining the member's profile could be shared between both the user and system. Users can provides the their interest and expertise explicitly. Different mechanisms can be used to make the interests explicit; probably, the most known is by subscriptions. Subscriptions allow users to specify, for example, which k-artefact of the shared-KR are of their interest. According to expertise a similar mechanism can be used. In this approach, the evolution of the member's profile depends on the user. On the other hand, the groupware system may play a useful role in the evolution of the member profile. Therefore, the groupware system requires of a component which may be in charge of observing users activity and update the member's profile with new interests and expertise. This component, users miner, has to be able to mine users actions in order to learn about new interests and expertise, and thus, to be able to update members' profiles. Actions mining can be done by the use of some data mining technique. They can discover new interest by "rating" those k-artefact that have be involved in the user activity. This may be complemented with the discovery of behavioral patterns that were mentioned in the previous subsection (5.4.1). For example, if a user is strongly involved in a discussion activity, it is possible to deduce the interest of the user in the discussed k-artifact. There exist many possible user miners to develop, because a user miner is strongly related to the kind of knowledge to discover. There are different techniques that can be applied. For example, in Bouthier' approach, he has uses neural networks to represent the evolution of the user's profile, and thus, to be able to deliver contextualized notifications. The member's profile is also used to implement indirect externalization, because it can works as a filter of the k-artifact to transfer.

Finally, an effective knowledge awareness mechanism also has to provide a suitable visualization of the delivered knowledge awareness information. It is not my aim to discuss visualization technique at this point, but I would only like to highlight which requirements of visualization that knowledge awareness states are. The knowledge awareness mechanism mainly delivers low-level and high-level notifications, it is in charge of indirect externalization and it is even responsible for highlighting differences between private and shared versions of the both knowledge repositories. Different approaches can be taken

to cover each visualization need. In Chapter 6, some examples of visualization will be introduced.

5.5 Conclusion

The third component of the ks-frameworks is the knowledge awareness. In this dissertation, I have both identified and defined knowledge and discussion awareness as the two specific awareness requirement for CKS systems. Awareness is strongly required because it is the means by divergence can be accepted since it complements the support of the divergence occurrences. Keeping people aware of the ks-activity works as the basic stimulus that a ks-community needs to generate new knowledge; and consequently, to keep the community "in action".

To design a knowledge awareness conceptual framework, I have made an analysis to discover which information is necessary to be tracked and captured when ks-activity occurs and how this information may be useful to the user. I have analyzed the possible questions that can be asked by the users and the different perspectives from they may make these questions. To catch up with knowledge activity, individuals need information that allows them to answer the questions: what, where, when, who, how and why and to track them from different perspectives. These perspectives are: the *k-artifact perspective*, the *actor's perspective*, the *activity perspective* and *KR perspective*. This analysis was made to both knowledge awareness and discussion awareness. In case of discussion awareness, the emphasis was made in the occurrence of divergences. In case of the discussion awareness, the *discussion thread perspective* is also part of the this analysis.

After analyzing awareness information needs, I have paid special attention to the knowledge awareness mechanism as a software component of the CKS system that is responsible for providing knowledge and discussion awareness. The knowledge awareness mechanism focuses on two main functionalities: the gathering and delivering of awareness information. In this thesis, I have only recognized the requirements of awareness mechanism which designers have to be concerned with.

Chapter 6

The Co-Protégé Prototype

Contents

6.1	Co-Protégé	119
6.2	Working in Co-Protégé	122
6.3	Co-Protégé Awareness	126
6.4	Co-Protégé Model, Metamodel and Generic Ontologies	127

The results of this thesis were tested in a software application. I have been involved in the development of a tool to build a shared KR. This KR, as it was discussed in previous chapters, uses ontologies to represent the knowledge and supports the ks-activity. To develop this software application I have adapted the Protégé platform, in order to add knowledge sharing functionalities. The resulting tool, which is called Co-Protégé, is a platform to develop collaboratively an ontological KR where people can follow the ks-process through the use of private and shared workspaces, the support of knowledge discussion occurrences and knowledge awareness information. Co-Protégé will be presented in the 8th International Protege Conference [Diaz05].

In this chapter, I will only present the details about CO-Protégé, how it works and how it was developed in order to support these new requirements; however, readers can find a brief introduction to Protégé in Appendix B.

6.1 Co-Protégé

Co-Protégé is a set of plugins that extends Protégé-2000 [Gennari03] in order to support the ks-activity, as it was described in previous chapters.

Preserving the Protégé-2000 developing philosophy, Co-Protégé was developed to add functionalities to edit ontologies and knowledge bases in a collaborative fashion. Therefore, in Co-Protégé, Protégé-2000 becomes in a more suitable groupware application that support the ks-activity.

There is a clear difference in the modality of creation and edition of a shared ontology between Co-Protégé and Protégé. Due to in Co-Protégé people do not make a direct edition of the shared ontology, but they also change it by means of the publication of ontological artefact. Co-Protégé's users manage simultaneously two ontologies: the private ontology and the shared one. Private ontologies can be edited in the private workspace. Users work at private workspace as they are working in a stand-alone fashion. There are similar functionalities that in Protégé-2000 plus some groupware functionalities as some kind of awareness of the shared ontology, see sections 6.2 and 6.3. There is also a shared workspace where the shared ontology is updated through the publication of ontological and discussion artefact. Many knowledge sharing functionalities are provided through the definition of special tabs, for example there exist tabs to see the differences between the private and shared ontologies, the divergences, and others. Next Figure 6.1 is a snap-shot of the interface of Co-Protégé when a user has already logged into the system.

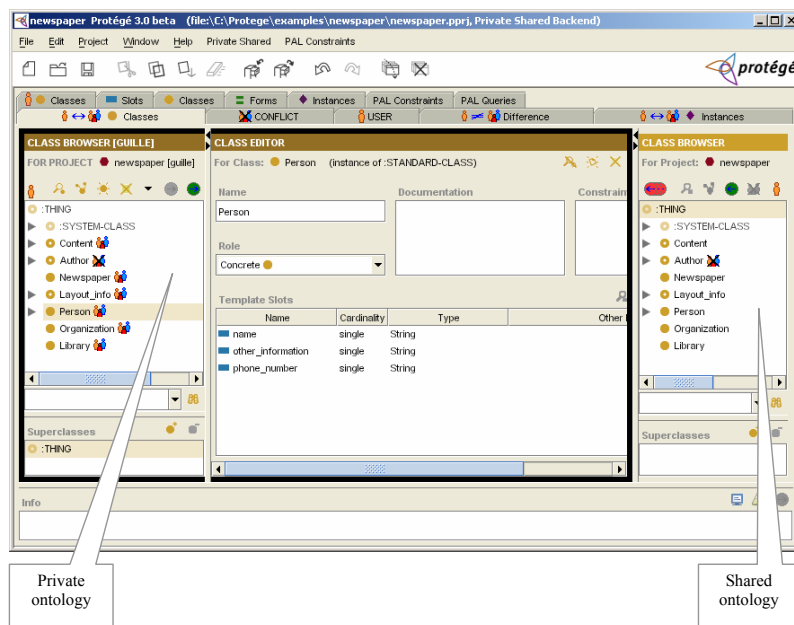


Figure 6.1: A snapshot of Co-Protégé. Both private and shared ontologies can be appreciated simultaneously. The black rectangle remarks the associated property pane to the current ontology. In this example, property pane shows the properties of the **Person** class from the private ontology.

Next sections will give details about Co-Protégé. The two following subsections will introduce the Co-Protégé architecture and the developed plugins and what a Co-Protégé project is and how it can be accessed. In section 6.2, it is described the underlying methodology to develop a collaborative ontological KR by using the two workspaces –private and shared. The section 6.3 provides a description of awareness mechanisms. Finally, the section 6.4 introduces the Co-Protégé's metamodel, model and generic ontologies.

6.1.1 Co-Protégé Architecture and Plugins

Co-Protégé extends Protégé through the definition of some plugins. The resulting architecture looks like any other Protégé extension because it follows the Protégé extension philosophy, which is described in section B.1. The Figure 6.2 is a schematic view of Co-Protégé architecture and its plugins. These plugins are:

- *Private-Shared Backend.* The *PrivateSharedBackend* is the plugin in charge of the ontologies storage management. It allows users the access to Co-Protégé ontologies, both the shared and the private ones. It follows the policies that are described in section B.1.
- *Co-Protégé tab-widget plugins.* They manage the Co-Protégé workspaces. They are in charge of the implementation of every tab described in section 6.2.
- *Co-Protégé slot-widget plugins.* Co-Protégé has also implemented some slot-widget plugins. For example, at the conflict tab, the list of argumentations is implemented as a Co-Protégé slot-widget plugin.

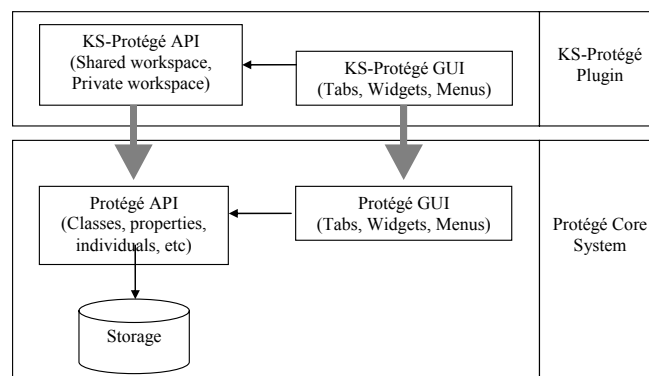


Figure 6.2: The Co-Protégé architecture.

6.1.2 Co-Protégé Project

In Co-Protégé a project is made up of the shared ontology plus every private ontology (one for each user). Both kinds of ontologies are stored in Standard Text File format. Although, at the moment, Co-Protégé does not support other formats, it is easy to extend with new formats.

Like Co-Protégé is a Protégé-2000 client-server application, a Co-Protégé project is defined following Protégé philosophy. A Co-Protégé project is a Protégé metaproject. In this metaproject every ontology is defined (the shared and each private) and the access permissions. Every user access to two ontologies, the shared and her/his private one.

6.1.3 Accessing to Co-Protégé project

Users log-in to a Co-Protégé project is like they were working in multi-user mode in Protégé-2000. People log in from a client machine to the Co-Protégé server. After that, the system opens the shared ontology and a private ontology. The private ontology is chosen based on user permissions; there is only one ontology with "read and write" access permission to each user.

6.2 Working in Co-Protégé

Co-Protégé imposes its own manner of carrying out the ks-activity. People edit the private ontology in the private workspace and then, they can publish ontological artefact to the shared ontology. Edition at the private workspace is carried out as users were working in a stand-alone fashion in Protégé-2000. The shared ontology is manipulated in a shared workspace and its "edition" is carried out by means of publications.

Co-Protégé visualization conserve the philosophy of Protégé-2000, it works with different tabs, each one dedicated to one aspect of the ks-activity (see Figure 6.1). There are tabs for modelling the shared-private workspace, the conflict tab, the user tab and the difference tab. Next, details of these special tabs will be introduced.

6.2.1 Shared-Private Workspace Tabs

First of all, it is important to notice that in Protégé philosophy the workspace to edit the ontology is made up of three tabs: classes and slot tabs to edit the conceptual ontology representation and the instances-tab to populate the knowledge base. Therefore, if we want to preserve the philosophy of ontology manipulation of Protégé-2000, we should define tabs for each type of ontological artifact, but even for each ontology: private and shared. This is not so convenient because it implies that users should be much trained in moving among the jungle of tabs, with the overhead that means having in mind two different ontologies.







In order to solve previous objection, Co-Protégé provides a tabs that "overlap" both workspaces in the same tab. There is one tab for each kind of k-artefact (class, slot and instance) and each one shows the two ontology versions: the private and the shared. They are the classes-shared-private tab, the slots-shared-private tab and the instances-shared-private tab.

Therefore, users simultaneously have the visualization of both, the private and the shared versions, easily achieving to a direct manipulation of the two ontologies. Observing Figure 6.1, readers can see the classes-shared-private tab where class hierarchies of both ontologies are shown, one on the left (the private) and the other on the right (the shared). The black rectangle identifies the active workspace where the user is working. The central pane is sensible to the active ontology. For example, in this figure, the central pane gives details of the selected class (the *person* class on the private ontology). To alternate between both workspaces is easy, because they are visually overlapped; users only need to click on the wished ontology pane.

Only the private side of a shared-private tab has the same functionality that the Protégé-2000 to edit a single ontology; the shared side cancels them because the shared ontology is updated by publications.

In Co-Protégé publications go between both sides. There is a series of operations that allow making contributions from one side to the other. Publications are supported by the operations enunciated in table 6.1.. They are organized in two groups: publications from private to shared side and publications from shared to private side (transfers). Each one is detailed in the table.

Table 6.1.: Publication operations

	Operation name	Description
From private to shared	 Publish	it publishes the selected ontological artefact in the private ontology to the shared one, if and only if it is an augmentative contribution.
	 Update	It updates the shared ontology based on the selected ontological artefact in the private ontology if and only if it produces an augmentative contribution
	 Publish/Update Anyway	It is a partial Publish or Update operation; it only performs the publication or updating of the private ontological artefact which provokes an augmentative contribution. It ignores the remainders. It is uses in the case a Publish or Update operations fail.
From shared to private	 Publish/Update	It publishes/updates the selected ontological artefact in the shared ontology to the private one , if and only if it is an augmentative contribution.
	 Publish/Update Anyway	It publishes/updates without making the contribution checking. It is responsibility of the user rearranging it private ontology.
	 Rebuild	It synchronizes the private ontology with shared version. If the operation provokes some incompatibility, shared ontological artefact prevail.

Co-Protégé system allows contributions only if the publication is augmentative. Checking is made following the rules enunciated in Chapter 4, section 4.5. Whatever may be the checking result, Co-Protégé informs this result at the bottom the shared-private workspace tab. Figure 6.3 is a snap-shot of the a checking result where the different types of notification can be observed.

Besides, at the shared-private tab it is also possible to open conflicts. On the shared side users can a create a conflict; conflict-tab section goes in details.

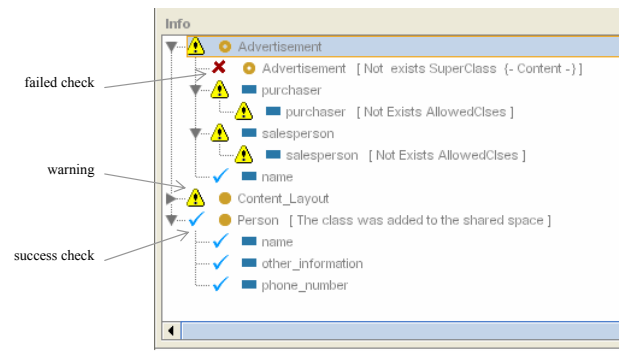


Figure 6.3: A snap-shot of checking results of a contribution. In particular it is an aborted contribution because the checking has failed.

User Tab

This is the tab to manipulate the user profile. Its look-and-feel is shown in Figure 6.4. User-tab is useful to manage user interest, that is, to define and observe some user's relationships with other knowledge elements. The dynamism of the user profile is managed directly by the user by updating user's interests and filiation. User interest can point to any kind of ontological artifact described by the metamodel of Co-Protégé, that is, elements of the shared domain ontology, other users, conflicts and conflict components.

There are some cases where the system is who changes the user's profile. The system is capable of tracking which ontological artifact were manipulated by the user, and in consequence, it can complete the user's interests. Then, this may be used to adapt delivered awareness information. Up to this moment, Co-Protégé provides a set of suggestions that help users to complete their profile [Baldo03]. For example, the "suggested interest" manager deduces new user's interest in based on users' activity. A technique of prediction based on doing mining of data over the collection of actions is used. Finally, it is the user who decides whether to incorporate the suggested interest to his/she user profile. These suggestions are written as queries to the knowledge base in PAL language. These set of predefined queries can be extended by users, they only need to write PAL queries and then every body has access to them.

Conflict Tab

A conflict is created in the shared-private tab by selecting the set of ontological artefact that will be put in conflict. After that, the ontological artifact are shown with the "in conflict" icon. To facilitate the visualization of conflict, it was decided to separate the conflict management form the shared-private tab. The conflict tab defines a space where users can browse and develop a conflict. Once a conflict was created, it becomes part of the conflict list, where all currently open conflicts are enumerated (see Figure 6.5). Users can add alternatives and argumentations. Alternative are created with ontological artifact from the private ontology. It is the mechanism that enables to publish contributions that did not pass the contribution checking.

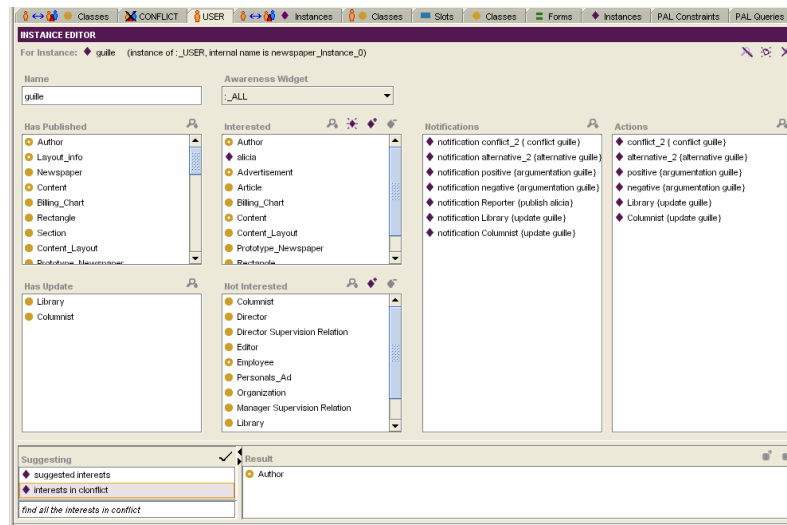


Figure 6.4: User tab.

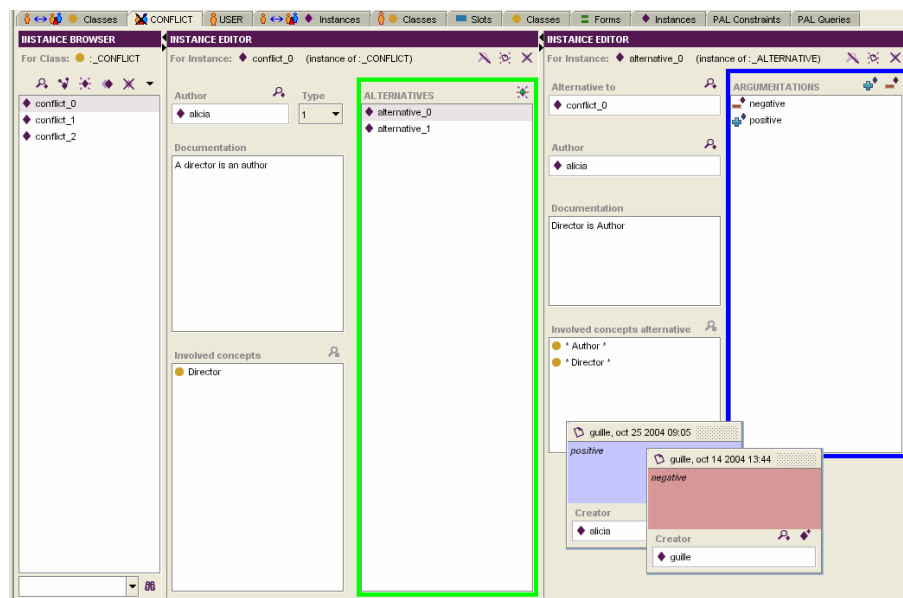


Figure 6.5: Conflict tab.

Difference Tab

It is a simple tab that shows differences between the private and the shared ontologies. The reason to design this tab was to help the users to find out coincidences and differences between its private ontology and the shared one. Figure 6.6 shows a snap-shot of this tab.

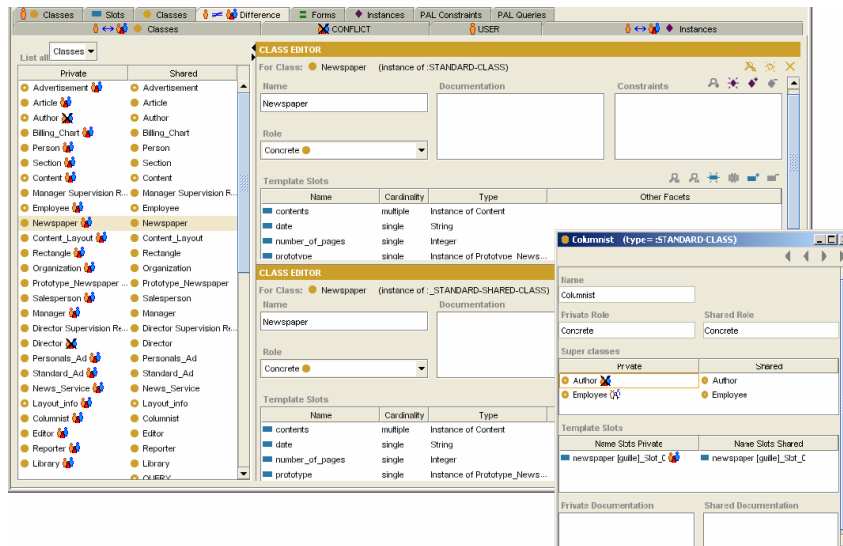


Figure 6.6: Difference tab.

More Tabs

Co-Protégé also support standard protégé tabs, but they are limited. They are mainly useful to browse the ontologies and the knowledge base. The private ontology supports every Protégé-2000 tab. However, with shared ontology the activity is restricted to browsing and conflict opening.

6.3 Co-Protégé Awareness

Co-Protégé provides a simple awareness. It offers awareness information about the ks-activity, focusing on the performed actions over the shared ontology and conflicts.

Knowledge awareness at Co-Protégé is managed through a mechanism of notifications. People constantly receive notifications about the ks-activity.

The Co-Protégé knowledge awareness mechanism tracks every contribution action. Contribution actions are gathered through the tracking of the performed actions at the shared-private tab and the conflict tab. Each time a new action takes place, Co-Protégé captures information about the performed action and creates an instance of the corresponding subclass of the `:_ACTION` class, being the action part of the knowledge base.

After gathering action information, the Co-Protégé awareness mechanism delivers this information through notifications. The system creates an instance of the `:_NOTIFICATION`

class. A notification is related to the action and attached to the users. If the user is on-line, immediately he/she is notified of the occurrence of a new event, otherwise he/she will be notified the next time he/she logs in.

The notification mechanism takes into account user's interests to deliver notifications. Only those notifications that answer the user's profile can be delivered. In the user's profile, people indicate what actions, ontological artefact or other users they are interested in.

Up to this moment, Co-Protégé supports two visualizations of notifications. One indicates the degree of similarity/difference that ontological artefact in both ontologies maintain. This is shown over the private ontological elements and looks as it is shown in Figure 6.7. This figure shows the private version of the ontology: The icon "both are the same" means that the private version of the current ontological artifact is exactly the same to which is in the shared version. This visualization is rendered each time any change occurs at the private or shared ontology. This visualization is useful to provide awareness of private divergence.

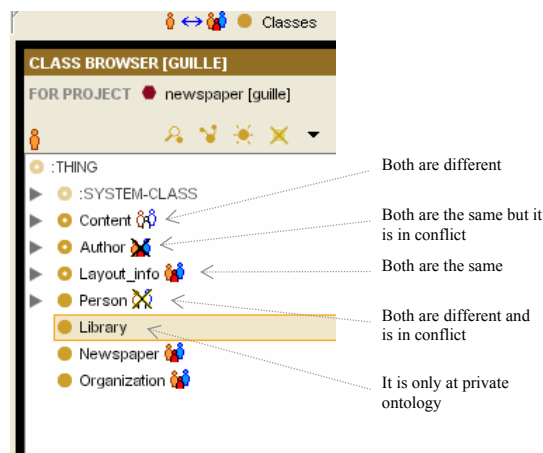


Figure 6.7: The visualization of the private divergence at the private side

The other visualization is more general and shows all the notifications in a chronological order (Figure 6.8). At the USER tab, users can specify different filter to show notifications.

6.4 Co-Protégé Model, Metamodel and Generic Ontologies

Co-Protégé uses the Protégé-knowledge model; then Co-Protégé ontologies use the same types of frames: classes, slots, facets and instances. However, it differs from the Protégé-2000 metamodel level. Co-Protégé use two different metamodels to model both ontologies: the private and the shared. Besides, Co-Protégé defines a set of generic ontologies to model

*	Class	Author	Involved	DateTime	Documentation
●	:_ALTERNATIVE-ACTION	◆ guille	◆ alternative_2	2005.02.10 15:18:53.210 GMT-03:00	
●	:_PUBLISH-ACTION	◆ alicia	● Reporter	2005.03.04 14:27:20.374 GMT-03:00	
●	:_UPDATE-ACTION	◆ guille	● Library	2005.03.04 14:29:03.733 GMT-03:00	
●	:_UPDATE-ACTION	◆ guille	● Columnist	2005.03.04 14:29:15.921 GMT-03:00	
●	:_CONFLICT-ACTION	◆ guille	◆ conflict_2	2005.02.10 15:18:40.742 GMT-03:00	
●	:_ARGUMENTATION-ACTION	◆ guille	◆ positive	2005.02.10 15:18:56.085 GMT-03:00	
●	:_ARGUMENTATION-ACTION	◆ guille	◆ negative	2005.02.10 15:18:56.898 GMT-03:00	

Figure 6.8: The visualization of the collection of notifications

the knowledge about the ks-activity. The Figure 6.9 shows how the Protégé's hierarchy of classes was extended to implement Co-Protégé. The new classes are in bold.

Any private ontology is considered as a Protégé-2000 project; therefore private ontologies respond to the regular Protégé-2000 metamodel; for example, if a user creates a *class DomainClass1* at his/her private ontology, *class DomainClass1* is an instance of **:STANDARD-CLASS**.

However, **:STANDARD-CLASS** is not enough to model an ontology artifact that is at the shared ontology. Any shared ontology artifact also needs to model other relationships that manifest features of being a shared artifact in a collaborative process. For example, a shared ontological artifact is strongly related to its creator or modifier (users) or it is needed to know when it was created. These features force to redesign the Protégé metaclass architecture to properly incorporate them. The following two subsections will deal with these problems.

6.4.1 The Shared Ontology Metamodel

Co-Protégé has its own metaclass architecture that is an extension of the Protégé-2000 metaclass architecture. It is done through the addition of a set of new metaclasses. Many of these metaclasses specialize the different Protégé-2000 metaclasses to model the primitive frames of a shared ontology. These new primitives are *shared-classes*, *shared-slots* and *shared-instances*. They are similar to the class, slot and instance frames defined in Protégé-2000, except they are extended to add the additional relationship that shared frames need to be a frame of a shared ontology.

A shared-class is a new template class that allows adding the needed attributes to extend the class with attributes derived from the ks-activity. Shared-classes are typed by the **:_STANDARD-SHARED-CLASS**⁵. This class is subclass of the **:STANDARD-CLASS**, inherits every **:STANDARD-CLASS** slots and adds the following slots (relationships):

:_PUBLISHED-BY slot: denotes the creator of the shared-class, who has published it at the shared ontology. It takes values in the generic class **:_USER**.

⁵ Co-Protégé class names begin with a ":_ " characters just for distinguish them from the Protégé-2000 class names that begin with the character ":".

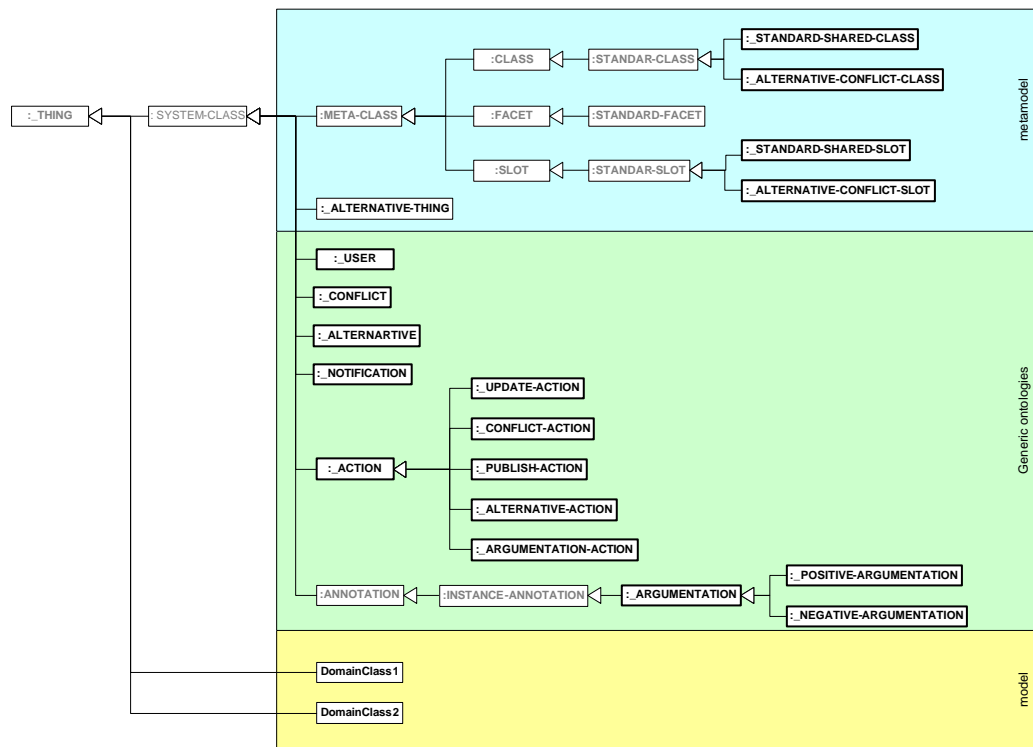


Figure 6.9: The Co-Protégé metamodel, model and generic ontologies.

- :_UPDATED-BY **slot**: denotes the set of modifiers of a shared-class, who have updated it (made augmentative contributions). It takes values in the generic class :_USER.
- :_IN-CONFLICT **slot**: denotes if a shared-class participates in a conflict, that is, if it is involved in some discussion thread. If it participates in a conflict, it takes value in the generic class :_CONFLICT.
- :_ACTIONS **slot**: denotes a set of actions performed over a shared-class. It takes value in the generic class :_ACTION.

Shared-slots are typed by the :_STANDARD-SHARED-SLOT. It is a subclass of the :STANDARD-SLOT. It also adds the same slots as a shared-class: :_PUBLISHED-BY, :_UPDATED-BY, :_IN-CONFLICT and :_ACTIONS. Notice that according to the semantic of a slot they can be applied independently of the source of the relationship.

The metaclass architecture of the shared ontology is used to model the ontological artifact that make up the element of the domain of discussion. This metaclass makes a difference between the ontological artefact of the private ontology and those of the shared one. Every ontological artifact of the shared ontology is model by :_STANDARD-SHARED-CLASS, :_STANDARD-SHARED-SLOT and :_STANDARD-SHARED-INSTANCE⁶, however the remainder of concept are model directly with the metamodel architecture of the Protégé-2000.

⁶The Co-Protégé prototype was developed only to manage the conceptual design of the shared ontology, still remains the treatment of instances.

6.4.2 Generic Ontologies of the Knowledge-Sharing Activity

Taking into account that the shared knowledge is not only knowledge about the domain of interest, but it is also knowledge about the ks-activity, the Co-Protégé needs to incorporate primitives that model these particular kind of knowledge. These new ontologies complete the shared ontology.

Knowledge about the ks-activity is independent of the specific domain of the discussion. The scheme of this knowledge should be applied to any environment that supports a ks-activity, whatever the domain of the interest of the community may be. Therefore, it is possible to design a generic ontology (Appendix A) to model the ks-activity.

Concepts like users, actions, conflict, argument, alternatives, and others are modelled by this set of generic ontologies. These generic ontologies are strongly related to the meta-class architecture of the shared ontology, as the reader has observed in previous section through the slots added to the `:_STANDARD-SHARED-CLASS`, `:_STANDARD-SHARED-SLOT` and `:_STANDARD-SHARED-INSTANCE` classes. The Figure 6.10 shows these generic ontologies and the relationships between them.

The generic ontologies to model the ks-activity gives the possibility to represent this activity as knowledge. This is the reason why it is possible to state that the Co-Protégé does not only manage knowledge about the domain, but also knowledge about the ks-activity. Knowledge about the ks-activity in some cases is automatically (implicitly) captured according to users actions at the workspace or in other cases it is provided explicitly by users.

Next, these generic ontologies will be introduced. They will be explained together with the relationships (slots) that they establish with other concepts. These classes are `:_USER`; `:_ACTION`, `:_CONFLICT`; `:_ALTERNATIVE-THING`; and `:_ARGUMENTATION`.

`:_USER` class models the profile of the users that participate in the ks-activity. A user is characterized by its filiation's information, the relationships that it establishes with ontological artefact of the shared ontology and with other users. This characterization is defined through the slots associated to the `:_USER` class. These slots are:

`:_NAME` slot: determines the identification of the user. There is a unique identification for each user.

`:_INTERESTED` slot: denotes the interest that the users have on ontological artefact of the shared ontology or on other users. It is a multiple slot that can contain instances of the `:_STANDARD-SHARED-CLASS`, `:_STANDARD-SHARED-SLOT` or `:_USER` classes. Through the definition of a slot constrains, interests can be refined or restricted (Protégé-2000 metamodel allows slots to have attached a constraint).

`:_NOT-INTERESTED` slot: it is the opposite to the `:_INTERESTED` slot. It allows making a explicit definition of non-interest.

`:_HAS-PUBLISHED` slot: denotes a multiple relationship with the ontological artefact that a user has published. It is a multiple slot that can contain instances of the `:_STANDARD-SHARED-CLASS` and `:_STANDARD-SHARED-SLOT`. It is the reciprocal slot to the `PUBLISHED-BY` slot which was defined at the `:_STANDARD-SHARED-CLASS` class.

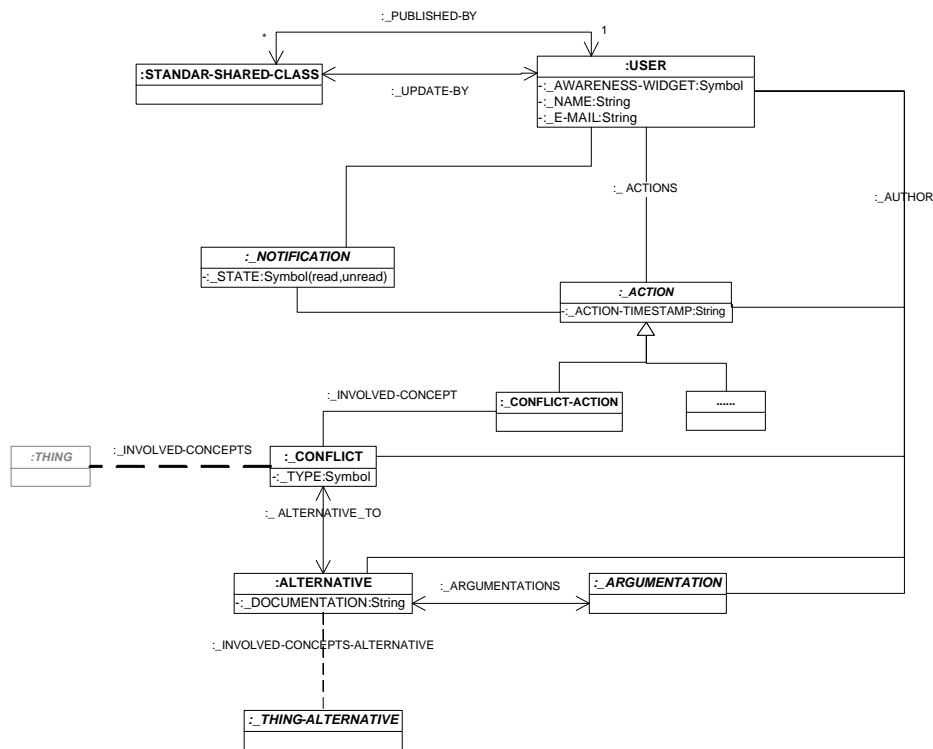


Figure 6.10: The Co-Protégé metamodel, model and generic ontologies.

:_HAS_UPDATED slot: it denotes a multiple relationship with the ontological artefact of the shared ontology that the user has updated. It is a multiple slot that can contain instances of the **:_STANDARD-SHARED-CLASS** and **:_STANDARD-SHARED-SLOT**. It is the reciprocal slot to the **:_UPDATED_BY** slot which was defined at the **:_STANDARD-SHARED-CLASS** class.

:_NOTIFICATIONS slot: denotes the received awareness information. It is an indirect multiple relation with the actions that have occurred at the shared workspace. Notifications are instances of the **:_NOTIFICATION** class and are a key element to support awareness information delivering; they will be explained in section 6.3.

:_ACTION class models the knowledge-sharing actions. These are the actions that have taken place over the shared ontology, like publication, open conflict, alternative contribution, argumentation, etc. **:_ACTION** class is an abstract class that is specialized in **:_PUBLICATION-ACTION**, **:_UPDATE-ACTION**, **:_CONFLICT-ACTION**, **:_ALTERNATIVE-ACTION** and **:_ARGUMENTATION-ACTION** classes. **:_PUBLICATION-ACTION** class model an initial contribution, **:_UPDATE-ACTION** class model an augmentative discussion contribution, **:_CONFLICT-ACTION** class models the open discussion action, **:_ALTERNATIVE-ACTION** class model a conflictive discussion contribution and **:_ARGUMENTATION-ACTION** class model a argumentative discussion contribution. Consuming knowledge actions are not supported at the prototypical version of Co-Protégé. Instantiation of the some subclass of **:_ACTION** class is responsibility of Co-Protégé system.

Action modelling is useful to preserve the history of the ks-activity that is essential

for the gathering of awareness information, but it is also useful to deduce the users' interests.

The structure of a `:_ACTION` class is defined by the slots:

`:_AUTHOR` slot: denotes who has performed the action. It is an instance of the `:_USER` class.

`:_INVOLVED-CONCEPTS` slot: is a multiple relationship with ontological artefact. They represent the ontological artefact involved in the action. They are instances of `:_STANDARD-SHARED-CLASS` and `:_STANDARD-SHARED-SLOT` classes or `:_CONFLICT`, `:_ALTERNATIVE-THING` and `:_ARGUMENTATION`.

`:_ACTION-TIMESTAMP` slot: denotes the time-stamp of the action.

`:_CONFLICT` class models the discussion thread. In Co-Protégé the ontological artefact of the shared ontology can be discussed. When a user opens a discussion, the discussion is opened over a set of ontological artefact. `:_CONFLICT` class slot structure allows supporting the discussion thread model presented in chapter 4, section 4.4.1, where the objected conceptualization is model by a `:_CONFLICT` instance and the relationships with the other discussion thread elements are represented through the slots:

`:_AUTHOR` slot: denotes who has opened the discussion. It is an instance of the `:_USER` class.

`:_INVOLVED-CONCEPTS` slot: is a multiple relationship with shared ontological artefact. These ontological artefact represent the *objected conceptualization* (see section 4.6.1). They are instances of `:_STANDARD-SHARED-CLASS` and `:_STANDARD-SHARED-SLOT` classes.

`:_CONFLICT-TYPE` slot: represents the *objected structure* (see section 4.6.1). At this prototype this slot is valued by a `:_SYMBOL` class instance that represent the type of objection, for example, the hierarchy structure between two shared-classes.

`:_ALTERNATIVES` slot: represents the *divergent contribution* attached to the *objected conceptualization*. It is a multiple relationship with `:_ALTERNATIVE` class instances.

`:_DOCUMENTATION` slot: is simply a string through which a user can contribute with a comment about the opened discussion. Notice that this slot is reused from Protégé-2000.

Close to the `:_CONFLICT` class are the `:_ALTERNATIVE-THING` and `:_ARGUMENTATION` classes. These three classes are responsible of keeping the discussion thread. Next, I will introduce them.

`:_ALTERNATIVE_THING` class models a divergent conceptualization of a shared ontological artifact. At the current version of Co-Protégé, alternatives to shared-classes and shared-slots are available. `:_ALTERNATIVE-THING` class is specialized in `:_ALTERNATIVE-CONFLICT-CLASS` and `:_ALTERNATIVE-CONFLICT-SLOT` classes to represent shared-classes and shared-slots respectively.

`:_ALTERNATIVE-CLASS` class is subclass of `:_ALTERNATIVE_THING` class and this last is subclass of `:THING`. `:_ALTERNATIVE_THING` identifies ontology artifact in the context of a conflict. It is necessary to differentiate the alternative ontological artifact of those already existing at the shared ontology. It is mainly due to the fact that alternative ontological artefact are allowed to have the same name of the objected one. For example, if one user opens a conflict to the shared-class **A**, then he/she can add to this class an alternative class also called **A**. The difference is that, while the original class **A** is instance of the `:_STANDARD-SHARED-CLASS`, the alternative class **A** is instance of `:_ALTERNATIVE-CLASS` class. `:_ALTERNATIVE-CLASS` class structure is given through the slots:

`:_AUTHOR` slot: denotes who has contributed with this divergent ontological artifact. It is an instance of the `:_USER` class.

`:_ALTERNATIVE-TO` slot: relates the alternative with its conflict owner. Its value in an instance of the `:_CONFLICT` class.

`:_INVOLVED-CONCEPTS` slot: it is a multiple relationship with ontological artefact. These ontological artefact represent the objected conceptualization and come from the private ontology. Alternative allows users to publish a new conceptualization that can be contributed as an augmentative contribution. They are instances of `:_STANDARD-SHARED-CLASS` and `:_STANDARD-SHARED-SLOT` classes.

`:_ARGUMENTATIONS` slot: collects in a multiple relationship its attached arguments. The arguments are instances of the `:_ARGUMENTATION` class.

`:DOCUMENTATION` slot: it is simply a string where the author can make a comment about the opened discussion.

`:_ARGUMENTATION` **class** models argumentations. `:_ARGUMENTATION` instances are attached to `:_ALTERNATIVE-THING`. `:_ARGUMENTATION` class is subclass of Protégé-2000's `:ANNOTATION` class. This class is an abstract class that is specialized in two subclasses: `:_POSITIVE-ARGUMENTATION` class and `:_NEGATIVE-ARGUMENTATION` class. Their structure is defined by the following slots:

`:_ANNOTATED_INSTANCE` slot: denotes which alternative the argumentation is attached to.

`:_ANNOTATION_TEXT` slot: is a text field to store the comments of the argumentation.

`CREATION_TIMESTAMP` slot: denotes time-stamp of the argumentation creation.

`:_AUTHOR` slot: denotes who has contributed with this argumentation. It is an instance of the `:_USER` class.

`:_NOTIFICATION` **class** models the delivered awareness information that user received. A notification is characterized by:

`:_NOTIFICATION-STATE` slot: indicates if the user has already seen the notification (take into account that users can work asynchronously). `:SYMBOL` and `:_READ` are valid values.

:_NOTIFICATION-USER slot: denotes the receiver of the notification. It is instance of :_USER class.

:_NOTIFICATION-ACTION slot: denotes the action to be notified. It is an instance of :_ACTION class.

Chapter 7

Conclusion

Contents

7.1	Thesis summary	135
7.2	Results	140
7.3	Future Work and Research	141

This chapter concludes the dissertation, and has three parts. Firstly, I make a summary of the thesis where I revisit the different chapters and remark their main goals and results. Secondly, I summarize the original contributions that this research has made to KM and CSCW research. Finally, I describe directions for future work based on the research done here.

7.1 Thesis summary

This dissertation has explored the occurrences of knowledge divergences in communities that shared knowledge as a design requirement for distributed groupware systems. The research was motivated by the general problem that sharing knowledge in a shared computational workspace seems forced, a little motivating and artificial. Furthermore, I observed that current knowledge sharing systems make the knowledge sharing activity difficult because they consider the knowledge sharing activity as a centralized accumulation of information rather than as process by means of which the group builds knowledge (develops a common understanding) and where discussion may take place. My perspective in this research has been that of a computer scientist and designer, and I have considered the support of a knowledge sharing activity and knowledge divergence occurrences with the goal of improving the conditions of usability of CKS systems.

Following research threads have been followed in this thesis. The primary research thread has concentrated on defining a knowledge sharing process which describes the knowledge sharing activity and considers also the discussion activity as part of it. The discussion activity states the problem of knowledge divergence occurrences. This knowledge sharing process has put forward special situations that will become requirements to CSCW field and groupware systems which are the object of the secondary research thread.

I have analyzed the knowledge sharing activity from both, KM field, and I have reached the conclusion that current approaches present weaknesses, especially when the knowledge sharing activity comprises occurrence of divergences. This has led us to the development of an alternative approach based on defining a knowledge sharing framework that supports the collaborative development of a decentralized KR by means of carrying out the ks-process, but also pushing the ks-activity.

When pursuing these research threads we have first reviewed the existing approaches from CSCW field to support ks-activity with divergence occurrence in Chapter 1.

In this chapter, I have presented the CSCW approach of the divergent management in groupware applications. From a general point of view, this approach considered divergences as a problem of coordination and articulation when people develop a computer-supported collaboratively activity. Particularly, it pays special attention to the problem of conflict occurrence because it is considered as a consequence of the interaction. But it is mainly considered as a problem of synchronization and versioning of the shared object. On the other hand, from a particular point of view, there are many groupware applications to support collaborative KR development, to develop knowledge sharing activity even with special emphasis on supporting a discussion activity. However, all of them only cover one or two aspects of the knowledge sharing activity. In this chapter I have also remarked the importance of the concepts: shared workspace and awareness. Because the former is where the collaborative activity takes place and the second one helps people to be aware of it.

The motivation for developing a novel CSCW approach to knowledge sharing activity has emerged by the analysis of the current approaches. In Chapter 2, that is in charge of introducing the knowledge sharing foundation and stating my approach of the knowledge sharing activity, I have introduced the requirements which have to be respected by a groupware application in order to support a suitable knowledge sharing activity. I have suggested, as a computer-support for a ks-community, a groupware application that is a CKS system that support the ks-activity, paying special attention to the discussion activity and the knowledge divergence occurrence. This system is characterized by the collaborative developing of a KR, by preserving the autonomy principle, the monotonic extension of the KR, and finally, the occurrence and coexistence of knowledge divergence.

The Chapter 2 is also in charge also of introducing the ks-process. The ks-activity is a collaborative learning process through which the community develops its own common understanding —knowledge repository. This process is a spiral process where knowledge goes emerging in each cycle. It describes an augmentative building of a KR through the contribution of "knowledge" and it is the means by knowledge is converted into tacit or explicit knowledge. People always add more knowledge in each contribution, whatever

this contribution means (augmentative or divergent). When the ks-process is computer-supported by the collaboratively development of the KR, this process describes how the knowledge is converted in two senses: tacit-explicit and private-shared. This conversion is carried out the next steps: externalization (from tacit to explicit knowledge and from private to private knowledge context), publication (from explicit to explicit knowledge and from private to shared knowledge context), internalization (from explicit to tacit knowledge and from shared to private knowledge context) and reaction which is the act of giving some kind of response to a previous contribution and is understood as an externalization/publication. Reaction is the means by divergences can arise. Another characteristic of the ks-process is that in each cycle it produces a new augmented knowledge version. Because publication is augmentative, each publication brings more knowledge to the shared knowledge context. Therefore, while the community is sharing knowledge, its knowledge context is constantly growing and in evolution.

After presenting here the different CSCW approaches to support ks-communities and foundations of this kind of communities, I introduced the knowledge sharing frameworks as a novel groupware approach to support the occurrence of knowledge divergence as part of ks-activity.

This framework provides a conceptualization of a groupware application which supports the ks-activity, in order to facilitate the ks-process by assisting the users to externalize, publish, internalize and, mainly, react before the knowledge "evolution". This framework describes the fundamental components of a CKS system. These components are: the knowledge-sharing workspace where the KR is built, the divergence management component which considers the occurrence of cognitive conflicts, and the awareness component that keeps people aware of the ks-activity. This framework is independent from the knowledge representation paradigm; the k-artifact models the minimal unit of knowledge which can be involved in the ks-activity.

The ks-workspace, which is introduced in Chapter 3, is a shared workspace that supports the collaborative development of a KR. This development is carried out through the conversion of the knowledge which is proposed by ks-process. Besides, the ks-workspace provides the mechanisms that comprise both individual knowledge and private activity, and shared knowledge and public activity. It also assists people to support private and public actions in a differentiated fashion, to enable alternating between shared and private KR and to allow exchanging knowledge between them, and isolates the knowledge representation system.

This approach also guarantees the monotonic extension of the KR because publications always provoke an augmentative version of the KR. Two approaches have been introduced herein (and they are also introduced in Chapter 3). One guarantees that any knowledge contribution always provokes a consistent shared version, in the sense that the contributed k-artifact can be "integrated" to the shared version without introducing any cognitive conflict. Otherwise, the contribution is not possible. For this reason, a mechanism for checking whether the contribution is augmentative is used. This mechanism is strongly dependent of the knowledge representation system. The other approach allows the occurrence of divergences, but they are expressed as a discussion contribution that is always an augmentative contribution. The discussion thread models the discussion ac-

tivity as a arrangement of discussion artefact. But also, the discussion thread, by means of the divergent artifact, enables users to publish cognitive conflicts without violate the monotonic extension of the shared version, because this artifact encapsulates the cognitive conflicts.

The ks-frameworks presented in Chapter 3, then, has been reconsidered in Chapter 4 because it is instantiated by a particular knowledge representation system. Ontologies paradigm was chosen to represent the knowledge. Due to its philosophical point of view, ontologies become useful as the knowledge representation system because the collaborative building of an ontology involves a collaborative design process to achieve to a conceptualization of the domain of interest. And thus, the ks-activity was reduced to the collaborative design of an ontological KR.

The ks-framework could be instantiated almost exactly as it was described in Chapter 3, except that the discussion activity was slightly changed. Particularly, at the discussion thread level, was possible to remark that in an ontological approach, it is not necessary to have an explicit representation of augmentative contributions because they can be managed implicitly. This is due to the kind of checking that was made to avoid the occurrence of non-augmentative contributions.

In this approach, ontologies were not only used to represent the domain knowledge, but also they were used to represent the members' profile and the carried out activity. This last kind of knowledge, which is represented by the ks-action and conflict ontologies, allow becoming the carried out ks-activity as part of knowledge of the community, and thus, they improve the quality of the accumulated knowledge. Besides, the capturing of the ks-activity knowledge is transparent to the users (there is no need of any explicit activity). However, the fact of having explicit knowledge about ks-activity will allow bringing better support to keep the group aware of this activity.

The third component of the ks-frameworks is the knowledge awareness. In this dissertation, I have both identified and defined knowledge and discussion awareness as the two specific requirement for CKS systems. Awareness is strongly required because it is the means by divergence can be accepted since it complements the support of the divergence occurrences. Keeping people aware of the ks-activity works as the basic stimulus that a ks-community needs to generate new knowledge; and consequently, to keep the community "in action".

Knowledge awareness has been defined as *the needed awareness information to keep a knowledge-sharing community up-to-date about the knowledge evolution*. Knowledge awareness plays a critical role when it comes to sharing knowledge, because it is a means to internalize and externalize knowledge. It helps people to notice new knowledge occurrences (contributions), and thus to raise users' curiosity. This is the clue to trigger a learning activity where individuals incorporate this public knowledge into their private knowledge context. Indirectly, pushing internalization is a way of pushing also the ks-activity, because this internalization becomes the seed of reaction occurrence. When people react, they are providing more knowledge, either augmentative or conflictive. This additional knowledge can emerges thanks to the stimulus received by the delivered knowledge awareness information. Therefore, knowledge awareness does not only work as an

engine of the ks-activity, but it also works indirectly as a source of knowledge. On the other hand, delivered awareness information has to be appropriated to the user needs.

I have define discussion awareness as a part of the knowledge awareness, which is more specific to take into knowledge discussion occurrences in the ks-activity. It is the required awareness to keep a knowledge-sharing community up-to-date about the evolution of the discussion . It is the kind of awareness in charge of making divergences acceptable. Because, it reinforces the occurrence of interaction among people. It comprises , together with the conflict occurrence, the means to improve interaction, and thus, the ks-activity.

Besides, the assisted internalization and to make the divergence evident are not the only contributions of the knowledge awareness. It can also be useful to aid people to externalize knowledge at their own individual-KR. It implies an indirect externalization at the individual-KR. This means that a new contribution to the shared-KR can be automatically incorporated to the individual one. Indirect externalization is complemented by the notifier component, which is in charge of giving awareness information about the occurrence of new contributions to the individual-KR. This is achieved by means of a local awareness mechanism which delivers information about changes at the individual knowledge context.

To design a knowledge awareness framework, I have made an analysis to discover which information is necessary to be tracked and captured when ks-activity occurs and how this information may be useful to the user. I have taken an analogous approach to the one in [Tam04] to understand the awareness information needs of the community members. I have analyzed the possible questions that can be asked by the users and the different perspectives from they may make these questions. To catch up with knowledge activity, individuals need information that allows them to answer the questions: what, where, when, who, how and why and to track them from different perspectives. These perspectives are: the *k-artifact perspective*, the *actor's perspective*, the *activity perspective* and *KR perspective*. This analysis was made to both knowledge awareness and discussion awareness. In case of discussion awareness, the emphasis was made in the occurrence of divergences. In case of the discussion awareness, the *discussion thread perspective* is also part of the this analysis.

After analyzing awareness information needs, I have paid special attention to the knowledge awareness mechanism as a software component of the knowledge sharing system that is responsible for providing knowledge and discussion awareness. The knowledge awareness mechanism focuses on two main functionalities: the gathering and delivering of awareness information. In this thesis, I have only recognized the requirements of awareness mechanism which designers have to be concerned with.

Although, the knowledge awareness framework was developed independently from the knowledge representation system, it might be applied to any virtual environment that supports a collaborative development of the KR which presents k-artefact as its building blocks. However, the use of ontologies to represent the knowledge has improved the usefulness of these kinds of awareness, since the knowledge in question is directly accessible.

The results of this thesis were tested in a software application. I have been involved in the development of a tool to build a distributed collaborative KR. This KR uses ontologies to represent the knowledge. It also allows users to develop the shared ontology

in collaboration, because it supports the collaborative edition of the shared ontology by following the approach presented in Chapter 3. To develop this software application, the Protégé platform was adapted, in order to add knowledge sharing functionalities. The resulting tool that is called Co-Protégé, is a platform to develop collaboratively an ontological KR where people can follow the ks-process through the use of private and shared workspaces, the support of knowledge discussion occurrences and a preliminary version of the knowledge awareness.

7.2 Results

We believe this thesis provides two major contributions to the management of divergence in ks-communities. The first is, certainly the integration of both collaborative activities: the KR development and the occurrences and coexistence of cognitive conflicts in the KR. Up to this moment, KM field uses groupware applications which have concentrated their attention on both activities, but in a dissociated way. Although, this field has used CSCW approach to support the ks-activity, the suggested groupware applications manage the discussion and the divergence occurrences outside of the collaborative design process. The novelty of this approach in this thesis is that the divergence occurrence is considered as part of the collaborative activity, because the discussion activity was coupled to the underlying collaborative activity. This has been defined the ks-process as a cycle process made up of externalization, publication internalization and reaction, as the process that models the knowledge sharing activity and convert the tacit knowledge in explicit. I have argued that the reaction step is the means through which discussion can take place and consequently, divergence can occurs.

Another characteristic of this approach is to consider the development of the shared-KR as an augmentative process, where users are always contributing with more knowledge, even whether this contribution is divergent. Any contribution is considered as the publication of an k-artifact. There are two main kinds of knowledge contributions, those that occur spontaneously and those that occur as part of a discussion. The last ones encapsulate the k-artefact that can complement or be divergent of an existing one. And, thus, cognitive conflict can take place at the shared-KR with out introducing semantic conflicts. This is a public divergence. In this thesis, I also propose a another approach to the occurrence of divergence, that is private divergence. It is the divergent that exists between the public and private versions. However, in this case, the divergence remains hidden in the private version.

There is a side effect of this primary contribution. This approach can be easily adapted to other groupware application that needs to manage the occurrences of divergences. Applications of collaborative design may be highly improved by integrating the occurrence of divergences to the collaborative design process, and by suitably replacing the concept of k-artifact by design artifact.

The second result that I have achieved is the recognition of special kind of awareness that makes divergences acceptable. I claims that users must be aware of the knowledge sharing activity to keep up-date-of the knowledge evolution. If there is no awareness users cannot be noticed about neither new knowledge occurrences or divergence knowledge

occurrence. This is the reason because I claims that special awareness is needed to assist to the internalization of knowledge and thus, the reaction may take place. Consequently, I have also recognized awareness as the source of knowledge and the engine of the ks-activity. I have defined knowledge and discussion awareness as the special kinds of awareness to assist to improve community interaction. While knowledge awareness is a awareness of the general ks-activity, discussion awareness pays special attention to the development of the discussion activity in order to detect and interpret the occurrences of divergences. The tracking of the ks-activity and mainly of those ks-actions that express divergences, is in charge of the mechanisms that implement knowledge and discussion awareness.

This thesis also gives two contributions to the field of ontology paradigm. On the one hand, I have give an alternative methodology to develop an ontology collaboratively. In this thesis, I have applied the ks-process to design and discussed a conceptualization of the domain of interest. The possibility of managing divergences occurrences is highly useful to ontologies designers, because the management of may alternatives is a common practice when the experts develops a shared ontology. As a consequence, the other contribution to the ontology field arise. I have developed Co-Protégé as a computer-support which extends Protégé capabilities in order to incorporate the results of this thesis.

7.3 Future Work and Research

There are some issues which derive from the research presented in this thesis. The problem of divergence occurrences can be faced in many other applications, mainly in those, where the collaborative activity involves the exchange of different perspectives. As mentioned in the previous section, one of the issues to investigate further concerns the application of these results to other distributed knowledge sharing systems, such project-support environment, collaborative business process engine, etc. Most of this application support the ks-activity as a parallel process to the subject of collaboration. It is important in this approach to study the integration of the ks-framework to existing collaborative infrastructures. This integration has to be done at the process level where both processes will be coupling. However, the semantic web is a more interesting field of application. Currently, the development of semantic portals has become an active area of research. It is reasonable to believe that the design of semantic portals may stem from the result of this thesis.

This research should be completed by a deep study of the impact of the ks-actions concerning to the consuming knowledge category. Although I have defined and placed the consuming actions in the ks-activity, it remains working on the context of querying the KR. The coexistence of knowledge divergence at the KR required some suitable query and deduction mechanisms in order to point to improve the development of the ks-activity. The first step in this thread is to define what a query and a deduction means in a KR that contains divergent knowledge occurrence. At the moment, the divergence is encapsulated in a discussion artifact, and traditional approaches to query and make deductions can be applied; however, it is interesting to intend to integrate this approaches with those coming from the Artificial Intelligence field and conflict reasoning algorithms.

The awareness component, which has been described in this thesis, is in an initial

state. In this research, I have identified knowledge and discussion awareness as suitable awareness, which helps users to recognize the existence of ks-activity, but mainly, the occurrence of divergence. Besides, the awareness information needs were found. However, it still remains to work in deducing high-level information and learning about the user's interest, and thus, to update the member's profile. In this thesis, I have suggested to take the approach of contextualize the gathered low-level information to derive high-level information; however I cannot still assure that this approach may be validated in real application context. Currently, the prototype only capture and store each ks-action in the knowledge database. Therefore, I already have a database of transactions, which is ready to be mined. At the moment, I only use the underlying querying language to make some simple deductions. Similarly, it may be done to update the member profile. Furthermore, works on visualization features should be done to provide a suitable knowledge awareness. In the prototype, I have deal with the visualization issues, but this work is relatively incipient.

Next, I will begin the evaluation stage in this research. Regarding the evaluation, there is one main evaluation to do. This is which allows me to conclude whether the results of this research, really, improve the usability of the current approaches. Before, the development of the prototype, I have already made a first informal experience where a group of experts in the domain of groupware applications has tried to design a domain ontology which represent its expertise. They have used the current version of the Protégé tool to design the ontology and they have also defined a Yahoo-group to discuss the design of the ontology. Based on this experience, I have drawn two conclusions. One is that people have problems to bring the conclusions about the design discussion to the ontology. Mainly, because nobody took the responsibility of transferring design conclusion to the shared ontology, and because they understand that it involves an extra effort. The other conclusion was that it is possible to get to the problem of conflict of interest and power. The current version of Protégé supports collaboration because it allows accessing to the shared ontology remotely, but that is all. Therefore, the design went forward and backward if there were divergences and users can not come to a consensus. Next step, it is to try the same experience but in the prototype. I plan to work with two groups of experts, that which has already made experience with Protégé, and another one with the same characteristics but this one has never used the prototype. Thus, I will have two results, one that will indicate if the ks-activity has been improved in a group by changing the approach, and the other one that will indicate if the application of the result of this research has improved the ks-activity independently of the group of users.

According to the prototype, next step is to adapt Co-Protege to OWL-Protege plugin to develop a suitable tool which will support the creation and development of semantic portals.

Appendix A

An Overview of Ontologies

A.1 Introduction

Ontology is an explicit specification of a conceptualization. An ontology provides a description of a particular viewpoint about a domain and that such a description must be explicit, in that it states a vocabulary for the domain, which is expressed by a certain degree of formality, and that a group commits to use the vocabulary according to the intended meaning associated with it in order to communicate. When the knowledge of a certain domain is represented in a declarative formalism, the set of objects that can be represented is called the universe of discourse. This set of objects, and the describable relationships among them, are reflected in the representational vocabulary with which the community represents its knowledge. In such an ontology, definitions associate the names of entities in the universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these terms. Pragmatically, a common ontology allows an ontological commitment for a domain of discourse. Ontological commitments are agreements meant to use the shared vocabulary in a coherent and consistent manner.

More formally, an ontology is a high level formal specification of certain knowledge domain: a *formal and explicit specification of a shared conceptualization* [[Gruber93], [Chandrasekaran99]].

- A conceptualization is an abstract, simplified view of the world, which is specified for some purpose. According to Genesereth and Nilsson [Genesereth87] a *conceptualization* is: *a triple consisting of universe of discourse, a functional basis set for the universe of discourse, and a relational basis set*. The universe of discourse is the set of objects on which the knowledge is expressed. The functional basis set groups a type of basic interrelationships among objects of the universe of discourse. A relational basis set is a set of a second kind of interrelationships holding among objects of the universe. We will denote a conceptualization as $\langle D, F, R \rangle$ where D represents the domain, that is the universe of discourse, F is the set of functional basis and where R is the relational basis set. For the purpose of this work, it might not be important to distinguish between the functional and the relational

basis set, in these cases we will denote a conceptualization as a simpler structure $\langle D, \mathbf{R} \rangle$ where \mathbf{R} is the set of all the interrelationships defined on the objects composing the universe of discourse. In this way, the conceptualization of a domain is a set of ontological descriptions $\{C_1, C_2, \dots, C_n\}$ where each C_i is an entity of the domain, a function or a relationship concerning one of the entities, that is $\forall C_i, i : 1..n, C_i \in D \vee C_i \in F \vee C_i \in R$. The explanation of each symbol C_i by assigning it a meaning corresponds to describing the domain according to a particular viewpoint and this viewpoint is the ontology.

- *Formal* refers to the fact that an ontology is a form of knowledge representation and has a formal software specification to represent such domain conceptualizations, i.e. an ontology has to be machine readable.
- *Explicit* means that all types of primitives, concepts, and constraints used in the ontology specification are explicitly defined. A body of formally represented knowledge is based on a conceptualization: the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that are held among them [Chandrasekaran99]. A conceptualization is an abstract, simplified view of the world, which is specified for some purpose.
- Finally, *shared* means that the knowledge embedded in ontologies is a form of consensual knowledge [Benjamins98], that is, it is not related with the individual, but accepted by a group [Vasconcelos01].

The meaning of the term ontology has different connotations in Philosophy and in Computer Science. [Guarino00] gave a characterization of the philosophical account for the term ontology as a particular system of categories accounting for a certain vision of the world [Guarino98]. In this perspective, an ontology is independent from the language used to describe it. However, the word ontology takes a different meaning in Artificial Intelligence, where it denotes an engineering artifact that is comprised of a specific vocabulary and of a set of explicit assumptions concerning the intended meaning of the words composing the vocabulary. Since the focus of this definition of ontology is the vocabulary, which is used to describe a specific reality, it is clear that the Artificial Intelligence notion of ontology is language dependent as opposed to the philosophical one [Tamma01].

In the context of a ks-activity, ontologies are proposed as the knowledge representation system; when the community externalizes its knowledge, it makes a conceptualization of its shared knowledge. A body of "formally" represented knowledge is based on a conceptualization: the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that are held among them [Chandrasekaran99].

The ontology is described in terms of a set of primitives [Gruber93]: there are abstract and physical objects in the world, such as a resource or a specific product; a set of objects denoting similar structure and behavior is considered a class; objects have properties or attributes that can have values, i.e. they can be represented as triplets (Object, Attribute, Value); for example, a specific person has a name, whose value could be 'Alan'; objects can exist in various relations with each other; for example, an organizational member belongs to a resource team; properties and relations can change over time; for example,

a corporate role assigned to a member may change over time. In [Gruber93], ontology is defined as the quintuple: (C, I, R, F, A) where:

- C is the set of the *concepts*, that is the set of the abstractions used to describe the objects of the world;
- I is the set of individuals, that is, the actual objects of the world. The individuals are also called *instances* of the concept.
- R is the set of relationships defined on the set C , where each $R \in R$ is an ordered n -ple $R = (C_1x C_2x \dots x C_n)$. For example **subconcept-of** is the pair (C_p, C_c) , where C_p is the parent concept and C_c is the child concept;
- F is the set of functions defined on the set of concepts that return a concept. That is, each element $F \in F$ is a function $F: (C_1x C_2x \dots x C_{n-1} \mapsto C_n)$.
- A : set of axioms, that is first order logic predicates that constrain the meaning of concepts, relationships and functions.

However, more recently, Sowa [Sowa00], CYC [Lenat90]() and Guarino [Guarino95] have proposed alternative upper ontologies. As a practical matter, a few could deny that there are objects in the world; these objects have properties that can take values; the objects may exist in various relations with each other; the properties and relations may change over time; there are events that occur at different time instants; there are processes in which objects participate and that occur over time; the world and its objects can be in different states; events may cause other events as effects; and objects may have parts. Further, perhaps not as basic facts of the world but as ways of organizing them, is the notion of classes, instances, and subclasses, where "*classhood*" is associated with shared properties. Thus, *is-a* relations indicating subclass relations are fundamental for ontology representations.

The representational repertoire of objects, relations, states, events and processes does not say anything about what classes of these entities exist. They are left as commitments to be made by the person modeling the domain of interest. Even at very general levels, such commitments already appear. Many ontologies agree on having the class "thing" or "entity" as root, but already at the next more specific level, they start to diverge, a fact which is clearly illustrated by the slightly different taxonomies of the top levels in existing ontology projects such as CYC, Wordnet, Generalized Upper Model, Gensim, etc. (see [Fridman-Noy97] for an overview). The more specific the domain to be modeled, the stronger the commitment of the ontology.

The ontologies presented in the literature can be classified according to different dimensions, which range from the level of generality of the concepts they describe [Guarino98], to the type of knowledge they model (be it related to the domain or the task)[vanHeijst97] and the degree of formality by which the terms and their meaning are expressed in the ontology [Uschold96]. The first dimension corresponds to the level of generality that is used in the description of the domain, and it is possible to distinguish the following types of ontologies. *Top-level ontologies*: this kind of ontology describes very general concepts

or common-sense knowledge such as space, time, matter, object, event, action, etc., which are independent of a particular problem or domain. *Domain ontologies*: this kind of ontology describes the vocabulary related to a generic domain such as medicine or physics. *Task ontologies*: this kind of ontology describes the vocabulary related to a generic task or activity such as diagnosis or selling. *Application ontologies*: this kind of ontology describes concepts depending both on a particular domain and on a particular task. They are often a specialization of both domain and task ontologies and correspond to the roles played by domain entities when they perform certain activities.

According to Van Heijst ontologies can be classified into two dimensions, which are the *amount and the type of structure of the conceptualization* and the *subject of the conceptualization*. *Amount and type of structure of the conceptualization*: This dimension is mainly concerned with the level of granularity of the conceptualization and thus can be subdivided into: *Terminological Ontologies* (just lexicons that specify the terminology), *Information Ontologies* (for example, database schemata) and *Knowledge Modelling Ontologies* (they specify conceptualizations of knowledge) *Subject of the conceptualization*: This dimension concerns the type of knowledge that is modelled in the ontologies. Four categories are distinguished along this dimension: *Application Ontologies*: specify those concepts that are necessary in order to model the knowledge required for a specific applications. Usually, application ontologies specialize terms taken from more general ontologies such as the domain and the generic ontologies described below and may extend generic and domain knowledge by representing method and task-specific components. Application ontologies are not reusable, they reuse knowledge which may be modelled in ontology libraries by tuning it for the specific application at hand. *Domain Ontologies*: specify those concepts that are specific of a particular domain. *Generic Ontologies*: specify concepts that are generic across many fields. Concepts in the domain ontologies may specialize in those in the generic ontologies in order to tune them to a particular domain. Generic ontologies correspond to the top-level ontologies in Guarino's classification. *Representation Ontologies*: explicate conceptualizations underlying knowledge representation formalisms. They provide a representational framework without making claims about the world, because they are meant to be neutral with respect to the world. Domain and generic ontologies are described by means of the primitives given in the representation ontologies.

Finally, ontologies can be classified into lightweight and heavyweight ontologies, depending on the degree of formality used to express them. Heavyweight ontologies are those which are provided with axioms, inference mechanisms aimed to equip ontologies with deductive power (e.g., inheritance), and that are characterized by a high degree of formality (e.g., underlying formal semantics). Lightweight ontologies, on the other hand, are those ontologies that define a vocabulary of terms with some specification of their meaning [Uschold98]. Ontologies differ also in the degree of formality by which the terms and their meaning are expressed in the ontology. Here, the knowledge expressed in the ontology might be the same, but they differ in the way in which it is expressed. *Highly informal*: are those ontologies expressed in natural language. Term definitions might be ambiguous due to the inherent ambiguity of natural languages. *Semi-informal*: these ontologies are expressed in a restricted and structured form of natural languages. Restricting and structuring natural language achieves improvement in clarity and reduction in ambiguity. *Semi-formal*: these are ontologies expressed in artificial languages, which

are formally defined, such as Ontolingua [Farquhar97]. *Rigorously formal*: these are ontologies whose terms are precisely defined with formal semantics, theorems and proofs of desired properties such as soundness and completeness.

In the last years, the number of environments and tools for building ontologies has grown exponentially. These tools are aimed at providing support for the ontology development process and for the subsequent ontology usage. The most relevant are Ontolingua [Farquhar96], Ontosaurus [Swartout97], and WebOnto [Domingue98]. The main similarity among these environments is that all of them have a strong relationship with a specific language (Ontolingua, LOOM and OCML, respectively). Actually, they were created to facilitate browsing and editing of ontologies in those languages. Furthermore, they were strictly oriented to research activities and most of them were built as isolated tools that did not provide many extensibility facilities. In the last years, a new generation of ontology-engineering environments have been developed. They have been created to integrate ontology technology in actual information systems. As a matter of fact, they are built as robust integrated environments or suites that provide technological support to most of the ontology lifecycle activities. They have extensible, component-based architectures, where new modules can be easily added to provide more functionality to the environment. Besides, the knowledge models underlying these environments are language independent. Among these environments, we can quote Protégé 2000 [Noy00a], WebODE [Arpirez01] and OntoEdit [Sure02].

Finally, with the huge emergence of the Semantic Web, tools for the development of DAML+OIL and RDF(S) ontologies have proliferated. In fact, the previous suites (Protégé 2000, WebODE and OntoEdit) allow importing and exporting DAML+OIL and RDF(S) ontologies. There are also several isolated tools that create DAML+OIL ontologies from different perspectives; the most representative are: OIEd [Bechhofer01] (a DL based tool), and DUET [Kogut02] (a UML-based plugin for Rational Rose). Recently, in 2001, the W3C formed a working group called Web-Ontology (WebOnt) Working Group. The aim of this group was to make a new ontology markup language for the Semantic Web, called OWL (Web Ontology Language). They have already defined a list of main use cases for the Semantic Web, have taken DAML+OIL features as the main input for developing OWL and have proposed the first specification of this language [Dean02].

Appendix B

A Brief Introduction to Protégé

B.1 Introduction

Protégé is an integrated software tool used by system developers and domain experts to develop knowledge-based systems [Protégé]. It is an extensible, platform-independent environment for creating and editing ontologies and knowledge bases. It is a tool which allows the user to construct a domain ontology, through the definition of classes, class hierarchies, slots and relationships between classes, instances, etc. The Protégé-2000 version will be presented here, and for simplicity reasons, in the remainder of this chapter, it will be referred to as Protégé.

Applications developed with Protégé are used in problem-solving and decision-making in a particular domain. Nowadays, Protégé can be used in those applications where concepts can be modelled as a class hierarchy. It allows also entering data, there is a special tab to define instances; and customizing data entry forms by rearranging and changing the fields of the default form.

Besides, Protégé also provides a platform which can be extended with graphical widgets for tables, diagrams, animation components to access other knowledge-based systems embedded applications; and a library in which other applications can be used to access and display knowledge bases.

Protégé tool⁷ accesses all of these parts through a uniform GUI (graphical user interface) whose top-level consists of overlapping tabs for compact presentation of the parts and for convenient co-editing between them. This interface permits an integration of (1) the modeling of an ontology of classes describing a particular subject, (2) the creation of a knowledge-acquisition tool for collecting knowledge, (3) the entering of specific instances of data and creation of a knowledge base, and (4) the execution of applications. The ontology defines the set of concepts and their relationships. The knowledge-acquisition tool is designed to be domain-specific, allowing domain experts to easily and naturally enter their knowledge of the area. The resulting knowledge base can then be used with a problem-solving method to answer questions and solve problems regarding the domain. Finally, an application is the resulting product created when the knowledge base is used in solving an end-user problem employing appropriate problem-solving, expert-system,

⁷ Protégé is available as free software under the open-source Mozilla Public License.

or decision-support methods. Finally, applications on top of these components are also executed within the integrated Protégé environment.

B.1.1 Protégé knowledge model

The knowledge model of Protégé is frame-based: frames are the principal building blocks of a knowledge base. Protégé *ontology* consists of classes, slots, facets, axioms and instances. The frames corresponding to the conceptual ontology level are: classes, slots, facet, axioms and templates and own slots.

Classes in Protégé are concepts in the domain of discourse and constitute a taxonomic hierarchy. Protégé supports multiple-inheritance: one class can have more than one superclass. The subclass relation is visualized in a tree where multiple inherited classes are denoted with a special icon. The root of the class hierarchy is the built-in class `:THING`. Both individuals and classes themselves can be instances of classes. A metaclass is a class whose instances are classes and they are modelled through the `:METACLASS` class.

Slots describe properties or attributes of classes and instances. A slot itself is a frame. Slots are first-class objects; they are defined independently of any class. When a slot is attached to a frame in the user's ontology, it describes the properties of that particular frame. When a slot is attached to a frame, it can have a value.

Facets describe properties of slots. One way to specify constraints in allowed slot values is through facets. The constraints specified using facets include cardinality of a slot (how many values the slot can have), restrictions in the value type of the slot (for example, integer, string, instance of a class), minimum and maximum value for a numeric slot, and so on. Facets define restrictions on an attachment of a slot to a class frame.

Axioms specify additional constraints.

Template and own slot A slot can be attached to a frame in one of two ways: as a template slot or as an own slot. An own slots describe a property of a (class or individual) frame itself rather than properties of instances of that frame. Template slots describe properties of instances of a class. Own slots do not propagate to either subclasses or instances of the frame to which they are attached. Template slots get inherited as template slots to the subclasses, and they become own slots for instances. An individual instance can acquire own slots only by being an instance of a class that has those slots as template slots and a class can acquire own slots only by being an instance of a metaclass which has those slots as template slots.

On the other hand, there are instance frames to represent the concrete level of an ontology. A Protégé knowledge base includes the ontology and individual instances of classes with specific values for slots.

Protégé provides a GUI where each type of primitives can be manipulated through different tabs (class tab, slot tab, instance tab). There are also other tabs to make queries

to the knowledge base, and to customize the GUI. In Figure B.1, it is possible to appreciate the appearance of a traditional window in Protégé. In this figure, the tab *Class* is shown, on the left side the class hierarchy is shown, and on the right side there are details of the selected class *Newspaper*.

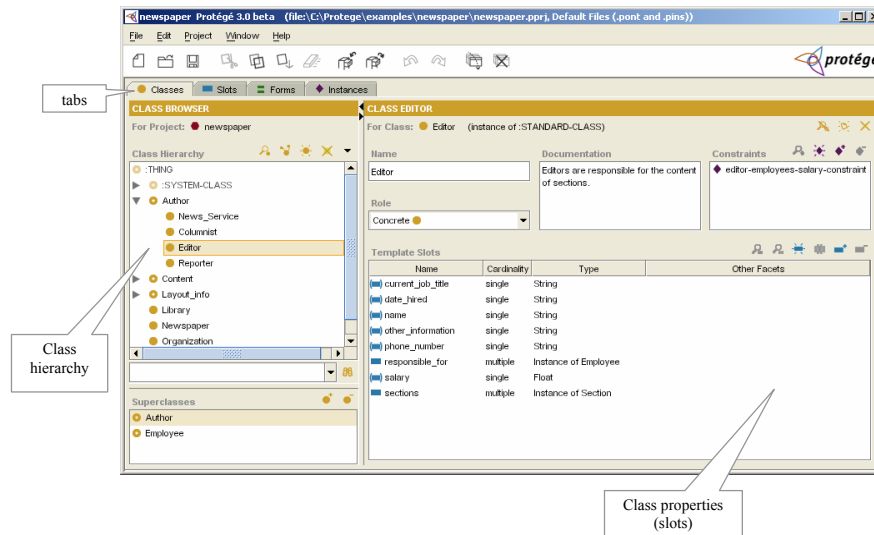


Figure B.1: A snapshot of Protégé-2000

Protégé's Metaclasses

Protégé supports certain kind of reflection, in the same address to some reflective computational system like Smalltalk, except for the fact that Protégé is a reflective knowledge representational system. Reflection in Protégé is based on the concept of metaclass.

A *metaclass* is a template for classes that are its instances. A metaclass describes how a class that instantiates this template will look: namely, which own slots it will have and what are the constraints for the values of these slots. The relation between a class and its metaclass is the same as an instance with its class. Own slots for a class—the slots that the class acquires from its metaclass—describe the properties of the class itself and not of its instances. For example, a class's role—concrete or abstract—defines whether or not the class can have direct instances. It is an intrinsic property of the class itself and not the property of its instances.

In Protégé, all metaclasses inherit from the system class: `:CLASS`, which inherits from `:META-CLASS` \rightarrow `:SYSTEM-CLASS`. By default, each class in Protégé is an instance of the `:STANDARD-CLASS` metaclass, which is a subclass of `:CLASS`. The `:STANDARD-CLASS` metaclass has template slots to store a class's name, documentation, a list of template slots, a list of constraints, and so on. These slots then become own slots for each of the newly created classes—instances of `:STANDARD-CLASS`.

The Protégé metaclass architecture enables one to manage class and instances in the same way, facilitating knowledge acquisition and ontology-editing process. Working at the ontological conceptual level is similar to working at the concrete ontological level,

because it is possible to customize and lay out the forms for specifying classes and slots in exactly the same way that it is possible to customize and lay out forms for acquiring instances.

Protégé allows users to extend the Protégé metamodel by defining their own meta-classes and to define new classes as instances of these user-defined meta-classes. This is the mechanism that was applied to extend Protégé-2000 to support collaborative ontology edition (details will be introduced in section ??).

Protégé-2000 built-in metaclass architecture

Protégé-2000 uses the metaclass mechanism to implement its own internal class structure. Meta-classes define the representation of all the frames in the system—classes, slots, facets, and instances. All the information about the frames, from name and documentation of a class to a list of its template slots and superclasses, is stored in the class's own slots. In other words, Protégé uses its own class structure to store the information about itself⁸.

The three Protégé system classes are `:CLASS`, `:SLOT` and `:FACET` and they serve as types for all classes, slots, and facets respectively. These classes do not have any template slots attached to them⁹. The Protégé-2000 knowledge model itself is implemented using the three standard subclasses of these classes: `:STANDARD-CLASS`, `:STANDARD-SLOT`, and `:STANDARD-FACET`. These three classes have the template slots that define the structure of their instances—classes, slots, and facets respectively.

`:STANDARD-CLASS` defines the default metaclass for classes. Template slots of this class define the standard own slots for classes. The slots store the class name, documentation, role, direct subclasses, direct superclasses, and direct template slots for the class, as it is shown in Figure B.2. Usually, all the user-defined meta-classes will be subclasses of `:STANDARD-CLASS`. The `:STANDARD-CLASS` is an instance of itself and therefore has the same sets of slots attached to it twice: once as own slots with values and once as template slots with value-type restrictions in the form of facets.

`:STANDARD-SLOT` defines the default metaslot in Protégé-2000. Template slots of `:STANDARD-SLOT` define the standard own slots for slot frames. These slots contain the slot name, its documentation, value type, numeric minimum and maximum, cardinality, and constraints. User-defined metaslots are subclasses of `:STANDARD-SLOT`.

`:STANDARD-FACET` defines the class to which all the built-in and user-defined facets belong. Currently there are no slots required for facets.

To summarize, the metaclass architecture in Protégé enables users to adapt and change the knowledge model of the system to suit the requirements of their domain and task. This is the main feature why I has decided to use Protégé to support the Prototype. It allows

⁸The users of Protégé-2000 do not need to see this internal information (and very few users do actually see it) unless they decide to explore the ontology describing the Protégé-2000 knowledge model.

⁹This feature allows Protégé-2000 developers to implement their own knowledge models in Protégé-2000 without making the same knowledge-model assumptions that Protégé-2000 does if they do not need it.

me to specialize `:STANDARD-CLASS` to adapt the prototype metamodel by incorporating features about collaborative edition of ontologies and discussion thread.

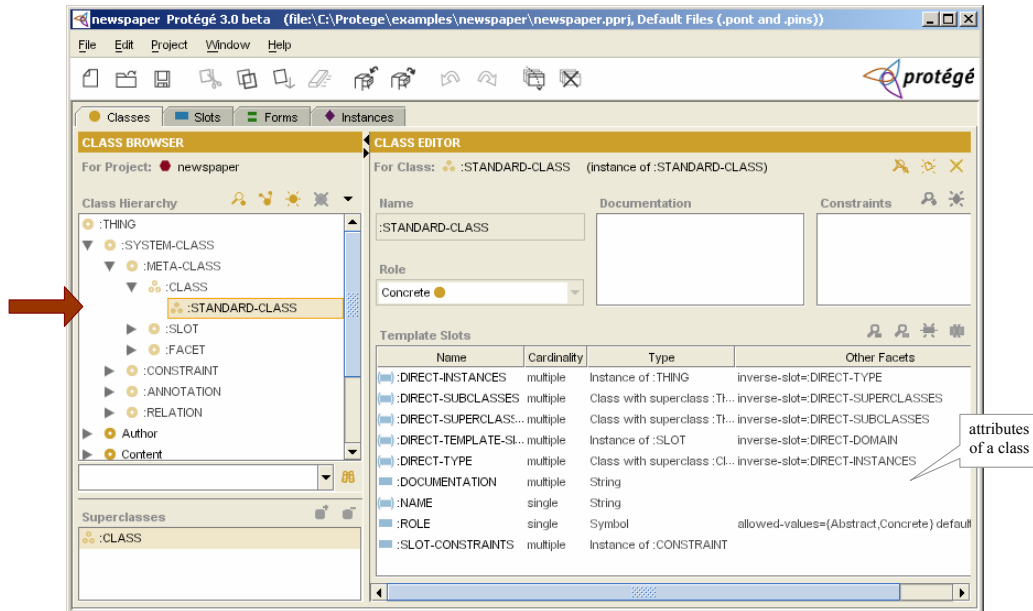


Figure B.2: Protégé-2000's class `:STANDARD-CLASS` is detailed

B.1.2 A Protégé Project

A Protégé project contains an ontology in a particular domain. It covers both the conceptual model and the concrete model, that is, all classes, slots and instances of a particular domain are included in a project.

A Protégé project is saved in a *pprj* (**P**rotégé **p**roject) file. This file contains the specific information about the Protégé-2000 interface. The project can be saved in different formats, but unless a special structure is needed (e.g., for exporting files), it should be selected Standard Text File format. It is possible to create, open and save the projects directly via the *pprj* file; there is no need to open any other file. With the *.pprj* file, internally, Protégé keeps two more files. These files contain further information about the ontology and instances of the project. When a *pprj* file is open, Protégé automatically loads these files. By default, the additional files are saved in text format: there are a text file containing classes and slots information, given the extension *pont* (**P**rotégé **o**ntology); and a text file containing the instances information, given the extension *pins* (**P**rotégé **i**nstances).

In addition to Standard Text Files (the default) format, Protégé also supports saving projects in JDBC Database, Resource Description Framework (RDF) and Web Ontology Language (OWL).

B.1.3 Extending Protégé

Plugins can be used to change and extend the behavior of Protégé. Protégé-2000 is itself written as a collection of plugins and these can be replaced individually or as a whole to completely alter the interface and behavior of Protégé. The Protégé API can be used directly by external applications to access Protégé knowledge bases and make use of Protégé forms without running the Protégé application.

A *plugin* is an extension of Protégé-2000. There are three basic types of plugins: *tab – widgetplugins*, *slot – widgetplugins*, and *backendplugins*. A *tab-widget plugin* is a user interface tab that appears in the main Protégé-2000 window beside the system tabs such as the classes tab. A *slot-widget plugin* appears on a form and it is used to view and acquire a value for a slot at an instance. A *backend plugin* is used to specify the mechanism that Protégé-2000 will use as storage (either as text or in a database).

B.1.4 Multi-user Mode

In addition to the stand-alone mode, Protégé-2000 can also run in a multiuser mode based on a client-server architecture (based on the Corba architecture). This mode supports a collaborative edition of an ontology.

Protégé specifies multi-user projects through the definition of a metaproject. The metaproject is a Protégé project that contains information about the projects to export and who can access them. This information is specified in an ontology that is instantiated when the sever is configured and updated when new projects and new users are incorporated to the multiuser environment.

The metaproject ontology (Figure B.3) is a simple model of users, security and projects. Instances of the `PROJECT` class will be made available to people identified with instances of the `USER` class. The "Guest" user instance models a default user. Every project has exactly one owner. People may become members of any number of groups. "World" is essentially a group that has everyone as a member. The security model is essentially the Unix file system security model. It manages security aspects (about who can access what) on top of whatever other security the underlying system provides (for example, with a firewall). Permissions are broken up into "read" and "write" access for users categorized into "owner", "group", and "world". The Unix security model is extended a bit in that individual users can be given specific access to a project.

The `PROJECT` class has instances of Protégé projects and these projects can be managed in a multiuser modality. These project instances and the metaproject must be located at the server machine.

To add both projects and users to a metaproject, the metaproject project is manipulated as a simple Protégé project. For example, to add to the multi-user version other projects that were already created in stand-alone mode, it is necessary create instances of projects at the metaproject. By default, only the owner has access to a project.

When a user opens a Protégé-client, it logs on to the server and can access to some ontology. Then in the client, a user can edit the remote ontology as it was a local project, except that any ontology updating is automatically replicated to other users. Protégé-2000 multi-user mode works with a persistence mechanism that avoids explicit saving.

Server has to be configured to specify the moment of autosaving.

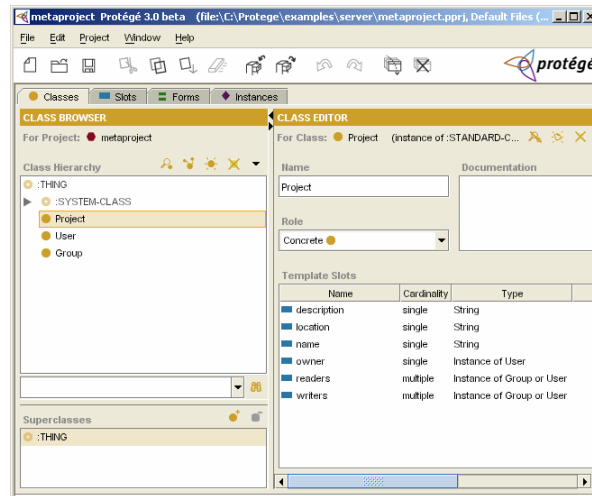


Figure B.3: The Protégé-2000 Project class

Protégé as a groupware application is rather limited. From the groupware point of view Protégé only supports the possibility of editing the ontology remotely. However, it does not support any coordination/articulation tool or synchronization mechanism or even awareness mechanisms.

Bibliography

- [Abecker99] Abecker, A. and Decker, S. *Organizational Memory: Knowledge Acquisition, Integration, and Retrieval Issues in Knowledge-Based Systems*. Lecture Notes in Artificial Intelligence, Vol. 1570, Springer-Verlag, Berlin, Heidelberg (1999) 113-124.
- [Agostini03] Alessandra Agostini, Sara Albolino, Giorgio De Michelis, Flavio De Paoli, Riccardo Dondi: *Stimulating knowledge discovery and sharing*. GROUP 2003: 248-257
- [Appelt99] Appelt, W. *WWW Based Collaboration with the BSCW System*. In Proceedings of SOFSEM'99, Springer Lecture Notes in Computer Science 1725, p.66-78; November 26 - December 4, Milovy (Czech Republic)
- [Arpirez01] Arpirez J.C. , Corcho O. , Fernandez-Lopez M. , Gomez-Perez A., *WebODE: a scalable ontological engineering workbench*, in: First International Conference on Knowledge Capture (KCAP01), ACM Press, Victoria, 2001, pp. 6-13.
- [Baldo03] Baldo G., Lemus M., Diaz A. 2003. Acciones como Fuente del Conocimiento de una Comunidad. Clei'03, La Paz, Bolivia, 29 Septiembre - 3 Octubre.
- [Bechhofer01] Bechhofer S. , Horrocks I., Goble C., Stevens R., *OilEd: a reason-able ontology editor for the Semantic Web*, in: Joint German/Austrian conference on Artificial Intelligence (KI01), Lecture Notes in Artificial Intelligence, vol. 2174, Springer, Berlin, 2001, pp. 396-408.
- [Benjamins98] V. Richard Benjamins, Dieter Fensel, Asunción Gómez-Pérez: *Knowledge Management through Ontologies*. PAKM 1998.
- [Birnholtz03] Jeremy P. Birnholtz, Matthew J. Bietz: Data at work: supporting sharing in science and engineering. GROUP 2003: 339-348
- [Boersma96] Boersma, J and Stegwee, R. (1996) Exploring the Issues in Knowledge Management, Accepted by the Information Technology Management in Europe track of the Information Resources Management Association International Conference.
- [Benerecetti00] Benerecetti M., P. Bouquet, and C. Ghidini. *Contextual Reasoning Distilled*. Journal of Theoretical and Experimental Artificial Intelligence, 12(3):279-305, July 2000.

- [Bonifacio02a] Bonifacio M., P. Bouquet e R. Cuel *Knowledge Nodes: the Building Blocks of a Distributed Approach to Knowledge Management*. Journal of Universal Computer Science, 8(6), 652-661, 2002.
- [Bonifacio02b] Bonifacio M., P. Bouquet and P. Traverso. 2002. *Enabling Distributed Knowledge Management. Managerial and Technological Implications*. Novatica and Informatik/Informatique, vol. III, n. 1.
- [Bonifacio03] Bonifacio M., Bouquet P., Mameli G., Nori M., 2003 *Peer - Mediated Distributed Knowledge Management*. Proceedings of AAAI Spring Symposium on Agent Mediated Knowledge Management (AMKM'03), Stanford University, March 24-26
- [Borges00] Borges, M.R.S., Pino, J.A. 2000. *Requirements for Shared Memory in CSCW applications*. Proceedings of WITS'2000, 10th Annual Workshop On Information Technologies and Systems, Brisbane, Australia, December, pp.211-216
- [Bouthier04] Bouthier, C. *Mise en Contexte de la Conscience de Group: Adaptation et Visualisation*. Doctoral thesis at the Institut National Polytechnique de Lorraine, France (2004).
- [Bouwer99] Bouwer, A., ArgueTrack: Computer Support for Educational Argumentation (1999). AI-Ed'99, Proceedings of International Conference on Artificial Intelligence in Education, 1999pp. 1-8
- [Brown95] Brown, J. S. and Gray. E. S. *The People Are the Company*. A Fast Company Magazine 1:78-82 (1995). Available at www.fastcompany.com/online/01/people.html, Februry, 2nd 2004.
- [Brown91] Brown, J.S. and Duguid, P. (1991). Organisational Learning and Communities of Practice: Toward a Unified View of Working, Learning and Innovation, *Organisation Science* 2(1) (p. 40-57).
- [Buckingham97] Buckingham S. 1997. Negotiating the Construction and Reconstruction of organizational Memories, *Journal of Universal Computer Science*, 2 (8), pp. 899-928.
- [Buckingham94] Buckingham S., Hammond N. 1994. Argumentation-Based Design Rationale: What Use at What Cost?, *International Journal of Human-Computer Studies*, , pp. 603-652.
- [Carstensen02] Carstensen, P. H. and K. Schmidt (2002). Computer supported cooperative work: New challenges to systems design. *Handbook of Human Factors*. K. Itoh. Tokyo, [in press].
- [Chandrasekaran99] Chandrasekaran, B., Josephson, J. and Benjamins V. *What Are Ontologies, and Why Do We Need Them?*. IEEE Intelligent Systems, Vol. 14, No. 1, (1999) 20-26.

-
- [Chalupsky00] Chalupsky H. *OntoMorph: A Translation System for Symbolic Knowledge*, in A. G. Cohn, F. Giunchiglia, and B. Selman, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Seventh International Conference (KR2000)*, Morgan Kaufmann Publishers, San Francisco, CA., 2000.
- [Clark96] Clark, H., 1996. *Using Language*, Cambridge University Press, Cambridge.
- [Cluts03] Marlin M. Cluts: *The evolution of artifacts in cooperative work: constructing meaning through activity*. GROUP 2003: 144-152
- [Conklin88] Conklin, J. and Begeman, M.L. *gIBIS: A Hypertext Tool for Exploratory Policy Discussion*. ACM Transactions on Office Information Systems, 4, 6, 1988, pp. 303-331.
- [Conklin01] Conklin J., A. Selvin, S. Buckingham Shum, M. *Facilitated Hypertext for Collective Sensemaking: 15 Years on from gIBIS*. Hypertext'01 Conference. (2001).
- [Corcho03] Corcho O., Fernández-López M., Gómez-Pérez A. *Methodologies, tools and languages for building ontologies: Where is their meeting point?* . Data Knowledge Eng. 46(1): 41-64 (2003)
- [Cunningham01] Cunningham, Ward and Leuf, Bo : *The Wiki Way. Quick Collaboration on the Web*. Addison-Wesley, ISBN 0-201-71499-X. (2001).
- [Davenport00] Davenport, T. H., Prusak, L. (2000). *Working knowledge: How organizations manage what they know*. Boston: Harvard Business School Press.
- [Davies03] Davies, J. Duke, A. and Sure, Y. *OntoShare - A Knowledge Management Environment for Virtual Communities of Practice*. Proc. of the 2nd International K-CAP, ACM Press (2003)20-27.
- [Dean02] Dean M., Connolly D., van Harmelen F., Hendler J., I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein, *OWL Web Ontology Language 1.0 Reference*, W3C Working Draft, 2002. Available from <<http://www.w3.org/TR/owl-ref/>>.
- [Diaz03] Diaz, A., Canals G. *Improving CoP Knowledge Sharing: a CSCW Approach Based on Awareness*. Proc. Caise'03 Forum, Eder and Welzer Eds. (2003)169-172.
- [Diaz04a] Diaz, A., Canals G. *Supporting Knowledge Sharing in a Community with Divergence*. Journal of Universal Computer Science. Proceedings of I-Know'04, 303-310.
- [Diaz04b] Diaz, A., Canals G. *Divergence Occurrences in Knowledge Sharing Communities*. LNCS, Springer Verlag -Proceedings of Criwg'04 17-24
- [Diaz05] Diaz, A., Baldo G. 2005. CO-Protégé: A Groupware Tool for Supporting Collaborative Ontology Design with Divergence. To be published in the Proc. of the 8th International Protégé Conference, July, 18-20th, Madrid, Spain.

- [Dignum02] Dignum V., *A Knowledge Sharing Model for Peer Collaboration in the Non-Life Insurance Domain*, Proceedings of German Workshop on Experience Management, 7. 8 March 2002, Berlin, Germany. Published in Lecture Notes in Informatics of the German Society for Informatics.
- [Dignum03] Dignum V., Dignum F., *Knowledge Market: Agent-Mediated Knowledge Sharing*. In: Proceedings of CEEMAS'03, 3rd International/Central and Eastern European Conference on Multi-Agent Systems, Prague, June 16 – 18, 2003.
- [Domingue98] Domingue J., *Tadzebao and Webonto: Discussing, Browsing and Editing Ontologies on the Web*, in: Proc. 11th Knowledge Acquisition Workshop (KAW98), Ban., 1998.
- [Drucker98] Drucker, P. (1998) *The Coming of the New Organisation*, Harvard Business Review on Knowledge Management, Harvard Business School Press, pp. 1-20.
- [Dourish92] Dourish P., Bellotti V. *Awareness and Coordination in Shared Workspaces*; Proc. ACM Conference on CSCW, ACM Press, New York (1992) 107-114.
- [Dourish95] Dourish P. *The parting of the ways: Divergence, data management and collaborative work* Proc. of the Fourth European Conference on CSCW, CSCW Mechanisms II, pages (1995) 215-230.
- [Easterbrook93] Easterbrook S. M., Beck E. E., Goodlet J., Plowman L., Sharples M., and Wood C. C. (1993) *A Survey of Empirical Studies of Conflict*. In S. M. Easterbrook (ed) *CSCW: Cooperation or Conflict?* London: Springer-Verlag, pp1-68.
- [Ellis91] Ellis, C. A., S. J. Gibbs and G. L. Rein (1991). *Groupware: some issues and experiences*. Communications of the ACM 34(1): 38-59.
- [Ellis93] Ellis, C. A., S. J. Gibbs and G. L. Rein (1993). *Groupware some issues and experiences*. Readings in groupware and computer-supported cooperative work. R. M. Baecker, Morgan Kaufmann Publishers, Inc.
- [Endsley95] Endsley, M., *Toward a Theory of Situation Awareness in Dynamic Systems*, Human Factors, 37(1), 32-64, 1995.
- [Farquhar96] Farquhar A., Fikes R., Rice J., *The Ontolingua Server: A Tool for Collaborative Ontology Construction*, in: Proc. 10th Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW96) Ban., 1996, pp. 44.1-44.19.
- [Farquhar97] Farquhar A., R. Fikes, J. Rice, *The Ontolingua Server: A Tool for Collaborative Ontology Construction*, in: Proc. 10th Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW96) Ban., 1996, pp. 44.1-44.19.
- [Fensel00] Fensel, D. (2000) *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, Springer-Verlag, Berlin.

-
- [Fridman-Noy97] Fridman Noy, N., Hafner, C. D. (1997). The state of the art in ontology design. *AI Magazine*, 18(3):53–74.
- [Greenberg96] Greenberg, S., 1996. Peepholes: Low Cost Awareness of One’s Community, *Proceedings of the Conference on Human Factors in Computing Systems (Conference Companion)*, Vancouver, 206-207.
- [Genesereth87] Genesereth, M. and Nilsson, N. J. *Logical Foundations of Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann Publishers (1987).
- [Gennari03] Gennari J., M.A. Musen, R. Fergerson. 2003. The Evolution of Protégé: An environment for knowledge-based systems development. *International Journal of Human-Computer Studies*, vol 58 (1): 89-123.
- [Gerosa01] Gerosa M., Fuks H., Pereira de Lucena J. 2001. Use of Categorization and Structuring of Messages in Order to Organize the Discussion and Reduce Information Overload in Asynchronous Textual Communication Tools. *CRIWG 2001*: 136-141
- [Grosso00] Grosso, W., Eriksson, H., Fergerson, R., Gennari, J., Tu, S., Musen, M. *Knowledge Modelling at the Millenium: the design and evolution of Protégé-2000*. Technical Report SMI-1999-0801, Stanford Medical Informatics, Stanford University (2000).
- [Gruber93] Gruber T. R. 1993. *Toward principles for the design of ontologies used for knowledge sharing*. Report KSL-93-04 Stanford Knowledge Systems Laboratory, USA (1993). Also in *Formal Ontology in Conceptual Analysis and Knowledge Representation*, edited by Nicola Guarino and Roberto Poli, Kluwer Academic Publishers (1994).
- [Grudin94] Grudin, J. (1994). *Computer-supported cooperative work: Its history and participation*. *IEEE Computer* 27(5): 19-26.
- [Guarino95] N. Guarino, M. Carrara, P. Giaretta, Ontologies and knowledge bases: towards a terminological clarification, in: N. Mars (Ed.), *Towards Very Large Knowledge Bases, Knowledge Building and Knowledge Sharing*, IOS Press, Amsterdam, 1995, pp. 25-32.
- [Guarino98] N. Guarino. 1998. Formal ontologies and information systems. In N. Guarino, editor, *Proceedings of FOIS’98*, page , IOS Press, Amsterdam.
- [Guarino00] Guarino N., Welty C.. 2000. *A Formal Ontology of Properties*. In, Dieng, R., and Corby, O., eds, *Proceedings of EKAW-2000: The 12th International Conference on Knowledge Engineering and Knowledge Management*. AAAI Press, Menlo Park. October, 2000.
- [Gutwin97] Gutwin, C. 1997. *Workspace Awareness in Real-Time Groupware Environments*. Ph.D. thesis, Department of Computer Science, University of Calgary, (Calgary Canada).

- [Gutwin02] Gutwin, C., Greenberg, S. *A Descriptive Framework of Workspace Awareness for Real-Time Groupware*. Computer Supported Cooperative Work, 11(3-4), Special Issue on Awareness in CSCW, Kluwer Academic Press (2002)
- [Hutchins90] Hutchins, E., 1990. The Technology of Team Navigation, in *Intellectual Teamwork: Social and Technological Foundations of Cooperative Work*, J. Galegher, R. Kraut and C. Egidio ed., 191-220, Lawrence Erlbaum, Hillsdale, NJ.
- [Hutchins95] Hutchins, E. (1995), *Cognition in the Wild*, Cambridge, MA, MIT Press.
- [Jacobson99] Jacobson, I., Booch, G., Rumbaugh, J. *The Unified Software Development Process*. Addison Wesley. ISBN 0-201-57169-2 (1999).
- [Johansen88] R. Johansen. *Groupware: Computer Support for Business Teams*. The Free Press, Macmillan Inc., New York.
- [Klein01] Klein, M. 2001. *Combining and relating ontologies: an analysis of problems and solutions*. In Gomez-Perez, A., Gruninger, M., Stuckenschmidt, H., and Uschold, M., editors, *Workshop on Ontologies and Information Sharing, IJCAI'01*, Seattle, USA.
- [Kogut02] Kogut P., Crane.eld S., Hart L., Dutra M., Baclawski K., Kokar M., Smith J., *UML for ontology development*, *The Knowledge Engineering Review* 17 (1) (2002) 61-64.
- [Kunz70] Kunz, W., and H. Rittel (1970) *Issues as Elements of Information Systems*. Technical report S-78-2, Institut fur Grundlagen der Planung i.A., Universitat Stuttgart, Keplerstrasse 11,7000 Stuttgart 1, Germany.
- [Lenat90] D.B. Lenat. Cyc: a large-scale investment in knowledge infrastructure. *Communications of the ACM*, volume 38, number 11, pages 33-38, November 1995.
- [Lorenz01] Lorenz, E. (2001), *Models of cognition, the Contextualisation of Knowledge and Organisational Theory*, *Journal of Management and Governance* 5: 307-330, 2001.
- [McGrath84] McGrath, J. E. (1984) *Groups: interaction and performance*, Engelwood Cliffs, NJ: Prentice- Hall. McGrath, J. E. (1991) *Time, Interaction and Performance (TIP): A Theory of Groups*. *Small Group Research*, Vol. 22, No. 2, pp. 147-174.
- [McGuinness00] McGuinness, Deborah L.. *Conceptual Modeling for Distributed Ontology Environments*. In *Proceedings of the Eighth International Conference on Conceptual Structures Logical, Linguistic, and Computational Issues (ICCS 2000)*. Darmstadt, Germany. August 14-18, 2000.

-
- [MacLean89] MacLean, A., R. M. Young, and T. P. Moran (1989) *Design Rationale: the argument behind the artifact*. Proceedings of CHI '89, pp. 247-252. New York: ACM.
- [Maedche03] Maedche A., S. Staab, N. Stojanovic, R. Studer, and Y. Sure. 2003. SEman-
tic portAL - The SEAL approach. In: *Spinning the Semantic Web*. D. Fensel, J.
Hendler, H. Lieberman, W. Wahlster (eds.), MIT Press, Cambridge, MA., pages
317-359.
- [McManus95] McManus, M. M. and Aiken, R. M., (1995). Monitoring computer-based
collaborative problem solving. *Journal of Artificial Intelligence in Education*. 4 6,
1995 pp. 307-336.
- [Molli01] Molli, P., Skaf-Molli H., Bouthier C. State Treemap: an Awareness Widget
for Multi-Synchronous Groupware. 7th International Workshop on Groupware
(CRIWG-01), Darmstadt, Germany, septembre 2001.
- [Muller99] Muller, H. and Shappert, A. (1999) The Knowledge Factory - A Generic
Knowledge Management Architecture, Workshop on Knowledge Management and
Organizational Memories, Proceedings of the Sixteenth International Joint Con-
ference on Artificial Intelligence (IJCAI99).
- [Nonaka94] Nonaka, I. *A dynamic theory of organizational knowledge creation*. Organi-
zation Science, 5(1)(1994)14-37
- [Nonaka95] Nonaka, I. and Takeuchi, H. (1995) *The Knowledge-Creating Company*, Ox-
ford University Press.
- [Nonaka98] Nonaka, I., Konno, N., 1998. *The concept of "Ba": building a foundation for
knowledge creation*. California Management Review, 40, 40-54.
- [Norman93] Norman, D., *Things That Make Us Smart*, Addison-Wesley, Reading, Mass.,
1993.
- [Noy00a] Noy N.F., Ferguson R.W., Musen M.A., *The knowledge model of Protégé-2000:
combining interoperability and flexibility*, in: 12th International Conference in
Knowledge Engineering and Knowledge Management (EKAW00), Lecture Notes
in Artificial Intelligence, vol. 1937, Springer, Berlin, 2000, pp. 17-32.
- [Noy00b] Noy, N.F. and Musen, M.A. (2000). PROMPT: Algorithm and tool for auto-
mated Ontology merging and alignment. In Proc. of the Seventeenth National
Conference on Artificial Intelligence (AAAI-2000), pp.450-455, Austin, TX:
AAAI Press.
- [Knublauch04] Knublauch, H., Ray W. Ferguson, Natalya F. Noy, Mark A. Musen The
Protégé OWL Plugin: An Open Development Environment for Semantic Web Ap-
plications Third International Semantic Web Conference - ISWC 2004, Hiroshima,
Japan (2004)

- [Patchen70] Patchen, M. (1970) Models of Co-operation and Conflict: A Critical Review. *Journal of Conflict Resolution*, Vol. 14, No. 3.
- [Pinto99] Pinto S., Gomez-Perez A., Martins J. 1999. *Some issues on ontology integration*. In Proceedings of the Workshop on Ontologies and Problem Solving Methods during IJCAI-99, Stockholm, Sweden, August.
- [Pinto04] Pinto S., Staab S., Sure Y., Tempich C. 2004. *OntoEdit empowering SWAP: A case study in supporting Distributed, Loosely-controlled and evolInG Engineering of oNTologies (DILIGENT)*. In Proc. of the 1st European Semantic Web Symposium, 10-12 May, Heraklion, Greece, Springer, LNCS.
- [Polanyi96] Polanyi, M. (1966) *The Tacit Dimension*, Garden City, NY, Doubleday.
- [Poltrock03] Steven E. Poltrock, Jonathan Grudin, Susan T. Dumais, Raya Fidel, Harry Bruce, Annelise Mark Pejtersen: *Information seeking and sharing in design teams*. GROUP 2003: 239-247
- [Pumareja03] Pumareja, D., Bondarouk, T., & Sikkel, K. (2003) *Supporting Knowledge Sharing Isn't Easy - Lessons Learnt from a Case Study*. Information Resource Management Association International Conference (IRMA'03), Philadelphia, May 2003.
- [Putnam87] Putnam, L. L., and M. S. Poole (1987) *Conflict and Negotiation*. In L. W. Porter, Ed., *Handbook of Organizational Communication: An Interdisciplinary Perspective*, pp. 549- 599, Newbury Park: Sage.
- [Reynolds04] Dave Reynolds, Paul Shabajee, Steve Cayzer. 2004. *Semantic information portals*. WWW (Alternate Track Papers and Posters), 290-291.
- [Reuber90] Reuber, A., Dyke, L., and Fischer, E. (1990) Using Tacit Knowledge Methodology to Define Expertise, Proceedings of the 1990 ACM SIGBDP conference on Trends and directions in expert systems, pp. 262-274.
- [Rittel73] Rittel, Horst and Melvin Webber (1973) *Dilemmas in a General Theory of Planning*, Policy Sciences 4, Elsevier Scientific Publishing, Amsterdam, pp. 155-159. Also Reprint No. 86, The Institute of Urban and Regional Development, University of California, Berkeley, California.
- [Schmidt96] Schmidt, K., Simone, C. (1996). *Coordination mechanisms: Towards a conceptual foundation of CSCW systems design*, *Journal of Computer Supported Cooperative Work*, vol. 5, no. 2-3.
- [Schmidt02] Schmidt K. (2002) *The Problem with 'Awareness' Introductory Remarks on 'Awareness in CSCW'* *Computer Supported Cooperative Work* 11(3-4), 285–298. Kluwer Academic Publishers.

-
- [Schreiber00] Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Robert de Hoog, Nigel Shadbolt, Walter Van de Velde and Bob Wielinga. 2000. Knowledge Engineering and Management: The CommonKADS Methodology, MIT Press, ISBN 0262193000.
- [Sowa00] Sowa, J. F. (2000). Knowledge Representation: Logical, Philosophical, and Computational Foundations, Brooks Cole Publishing Co., Pacific Grove, CA.
- [Staab04] Staab S., R. Studer (eds.). 2004. Handbook on Ontologies. International Handbooks on Information Systems, Springer Verlag,
- [Stahl99] Sthal, G. *WEBGUIDE: Guiding Collaborative Learning on the Web with Perspectives*. AERA '99, Montreal, Canada (1999)
- [Stahl05a] Stahl G. 2005. *Group Cognition in Computer Assisted Learning*. Journal of Computer Assisted Learning (JCAL).
- [Stefik87] Stefik, M., G. Foster, D. G. Bobrow, K. Kahn, S. Lanning, and L. Suchman (1987) *Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings*. Communications of the ACM, Vol. 30, No. 1, pp. 32-47.
- [Schrott03] Gregor Schrott, Daniel Beimborn: Managing virtual knowledge networks: topology and performance. GROUP 2003: 198-204
- [Suchman87] Suchman, L. (1987), *Plans and Situated Actions: the Problem of Human Machine Interaction*, Cambridge University Press.
- [Sure02] Sure Y., M. Erdmann, J. Angele, S. Staab, R. Studer, D. Wenke, *OntoEdit: collaborative ontology engineering for the semantic web*, in: First International Semantic Web Conference (ISWC02), Lecture Notes in Computer Science, vol. 2342, Springer, Berlin, 2002, pp. 221-235.
- [Suthers97] Suthers, D., Toth, E. and Weiner, A., (1997). An Integrated Approach to Implementing Collaborative Inquiry in the Classroom. Proc. Of CSCL'97, Computer Supported Cooperative Learning, Toronto, 1997, pp272-279
- [Swartout97] Swartout B. , Ramesh P., Knight K., Russ T. , *Toward Distributed Use of Large-Scale Ontologies*, AAAI Symposium on Ontological Engineering, Stanford, 1997.
- [Tam04] Tam, J. and Greenberg, S. (2004), *A Framework for Asynchronous Change Awareness in Collaboratively-Constructed Documents*. CRIWG'04 X International Workshop on Groupware, Lecture Notes in Computer Science (LNCS Number TBA), Springer Verlag (September 5-9, San Carlos, Costa Rica).
- [Tamma01] Tamma, V.A.M., 2002. *An Ontology Model supporting Multiple Ontologies for Knowledge sharing*. PhD Thesis, The University of Liverpool.

- [Tennison98] Tennison, J. and Shadbolt, N. R. 1998. A Tool to Support Living Ontologies. In Proceedings of Proceedings of the 11th Workshop on Knowledge Acquisition, Modelling Management. Gaines, B. R. and Musen, M., Eds.
- [Towell01] Towell J., Towell E. 2001. *Virtual Scientific Collaboration and Nonaka's Ba*. Proceedings of the 34th Hawaii International Conference on System Sciences.
- [Uschold96] Uschold M. and M. Gruninger. *Ontologies: principles, methods, and applications*. Knowledge Engineering Review, volume 11, number 2, pages 93–155, 1996.
- [Uschold98] Uschold M. Knowledge level modelling: concepts and terminology. Knowledge Engineering Review, 1998.
- [vanHeijst97] G. van Heijst, A.Th. Schreiber, and B.J. Wielinga. Using explicit ontologies in kbs development. International Journal of Human-Computer Studies, volume 45, pages 184-292, 1997.
- [Vasconcelos01] Vasconcelos J., C. Kimble, F. Gouveia, and D. Kudenko. *Reasoning in Corporate Memory Systems: A Case Study of Group Competencies*. Proc. of the 8th International Symposium on the Management of Industrial and Corporate Knowledge, France, October 2001, pp. 243-253. ISBN: 2-913923-03-8
- [Walsh95] Walsh, J. P. (1995). *Managerial and Organizational Cognition: Notes from a Trip Down Memory Lane*. Organizational Science 6(3): 280- 321.
- [Weber19] Weber, M. (1919), *Wirtschaft und Gesellschaft*, Trad. anglaise, Economy and Society, 1978, University of California Press, Berkley and Los Angeles, California.
- [Wenger98] Wenger, E. (1998), *Communities of practice: Learning as a Social System*, Systems Thinker, June.
- [Wenger02] Wenger E., McDemott R., Snyder W. 2002 *Cultivating Communities of Practice*. Harvard Business School Press.
- [Wenger01] Wenger E. 2001. Supporting Communities of practice. A Survey of Communities-oriented technologies. Research and Consulting. <http://www.ewenger.com/tech/index.htm>.
- [Wilson91] Wilson, P. (1991). *Computer supported cooperative work : an introduction*. Oxford, England Norwell, MA, Intellect ; Sold and distributed in the U.S.A. and Canada by Kluwer Academic Publishers.
- [Yu99] Yu, R. and Yam, S. C., (1999). An Intelligent Agent in Web-based Argumentation. Artificial Intelligence in Education. In S.P. Lajoie and M. Vivet Eds., IOS Press, pp 551-557.
- [Zacklad03a] Zacklad, M. *Un cadre théorique pour guider la conception des collecticiels dans les situations de coopération structurellement ouvertes*. In: BONARDI, C.,

GEORGET, P., ROLAND-LEVY, C., et ROUSSIAU, N. Psychologie Sociale Appliquée, Economie Médias Nouvelles Technologies. InPress: Paris, 2003. p. 135-164.

[Zacklad03b] Zacklad, M. *Communities of Action: a Cognitive and Social Approach to the Design of CSCW Systmes*. GROUP'2003, 09-12 November 2003, Sanibel Island. ACM, 2003, 190-197 Goodlet98