Méta-analyse pour le génie logiciel des systèmes multi-agents

Luis Alfonso Razo Ruvalcaba Razo Ruvalcaba

UNIVERSITÉ DE GRENOBLE

THESIS

to obtain the grade of

DOCTOR OF THE UNIVERSITY OF GRENOBLE

Speciality :**Computer Science**

Ministerial order :

Presented by

Luis Alfonso Razo Ruvalcaba

Thesis directed by **Michel OCCELLO**
and co-directed by **Félix Francisco RAMOS CORCHADO**

prepared within **Laboratoire de Conception et d'Intégration des Systèmes (LCIS) and the Guadalajara Campus of Centro de Investigación y Estudios Avanzados del I.P.N (CINVESTAV)**
and the **Ecole Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique (ED MSTII)**

Meta-analysis applied to Multi-agent Software Engineering

Thesis defended in public in 27 July 2012 ,
in front of the jury composed of :

Marie Pierre Gleizes Professeur - Université Paul Sabatier, Rapporteur
Joel Quinqueton Professeur - Université de Montpellier, Rapporteur
Luis Ernesto López Mellado Professor - CINVESTAV-IPN Guadalajara, Examiner
Víctor Hugo Zaldívar Carrillo Professor - ITESO - Instituto Tecnológico y de Estudios Superiores de Occidente, Examiner
Dominique Decouchant Chargé de Recherche CNRS - LIG CNRS et Universidad Autonoma Metropolitana (UAM), Examiner
Michel OCCELLO Professeur - Université de Grenoble II, Thesis Co-Director
Félix Francisco RAMOS CORCHADO Professor - CINVESTAV IPN Guadalajara, Thesis Co-Director

Acknowledgements

I would like to give thanks to the National Council on Science and Technology, CONACyT, for providing me the PhD Scholarship number 49178 and the Rhône Alpes Region to support part of the present work. This research is also partially supported by CoECyT-Jal project No. 2008-05-97094. I'd also like to thank my thesis supervisors, Pr. Michel Occello and Pr. Félix Francisco Ramos Corchado, for their invaluable guidance in the development of this research; their patiently treat and help in difficult moments, his perseverance and confidence in me and my work push me to the end of this thesis work. Thanks to my co-workers in the Distribuited Systems group at CINVESTAV and the COSY research group members at LCIS that always encourage me to go beyond. Special dedicatory to my parents, my father Luis Alfonso Razo Arias and my mother María Socorro Ruvalcaba Plasencia, my sisters Flor, Cristina and Estefania.

Thanks to my family, my wife Aleksandra and my son Maxim they are my greatest life motivation and source of inspiration.

Abstract

From a general point of view this thesis addresses an automatic path to build a solution choosing a compatible set of building blocks to provide such a solution to solve a given problem. To create the solution it is considered the compatibility of each available building block with the problem and also the compatibility between each building block to be employed within a solution all together.

In the particular perspective of this thesis the building blocks are meta-models and the given problem is a description of a problem that can be solved using software using a multi-agent system paradigm.

The core of the thesis proposal is the creation of a process based on a multi-agent system itself. Such a process analyzes the given problem and the available meta-models then it matches both and thus it suggests one possible solution (based on meta-models) for the problem. Nevertheless if no solution is found it also indicates that the problem can not be solved through this paradigm using the available meta-models.

The process addressed by the thesis consists of the following main steps: (1) Through a process of characterization the problem description is analyzed in order to locate the solution domain and therefore employ it to choose a list of most domain compatible meta-models as candidates. (2) There are required also meta-model characterization that evaluate each meta-model performance within each considered domain of solution. (3) The matching step is built over a multi-agent system where each agent represents a candidate meta-model. Within this multi-agent system each agent interact with each other in order to find a group of suitable meta-models to represent a solution. Each agent use as criteria the compatibility between their represented candidate meta-model with the other represented meta-models. When a group is found the overall compatibility with the given problem is evaluated. Finally each agent has a solution group. Then these groups are compared between them in order to find the most suitable to solve the problem and then to decide the final group.

This thesis focuses on providing a process and a prototype tool to solve the last step. Therefore the proposed path has been created using several concepts from meta-analysis, cooperative artificial intelligence, Bayesian cognition, uncertainty, probability and statistics.

Resumen

Desde un punto de vista general esta tesis aborda el problema de encontrar una solución a un problema determinado usando un conjunto de bloques de construcción. Esto se hace teniendo en cuenta la compatibilidad de cada uno de los bloques de construcción para resolver el problema y la capacidad de interacción entre estas partes para formar una solución juntos.

En la perspectiva particular de la tesis los bloques de construcción son meta-modelos y el problema dado es una descripción de un problema puede ser resuelto utilizando software y que hay que verificar si es posible resolverlo con una configuración basada en un sistema multi-agente.

El núcleo de la propuesta de tesis es un proceso que analiza un problema dado, verificando si es posible resolverlo usando un sistema multi-agente, si es así se sugiere una posible solución basada en sistema multi-agente para este problema. También puede indicar que el problema no puede resolverse a través de este paradigma o que no se cuenta con la información suficiente para determinar si sí o si no.

El proceso abordado por la tesis consta de los siguientes pasos principales: (1) A través de un proceso de caracterización de la descripción se identifican las características y el dominio de la posible solución para luego elegir un sub conjunto de posibles candidatos meta-modelos. (2) Los meta-modelos cuentan con caracterizaciones en varios dominios de solución. (3) Se crea un sistema multi-agente en el que cada agente representa a un candidato meta-modelo del subconjunto elegido en la primera fase. De esta sociedad de agentes estos interactúan entre sí para encontrar un grupo de meta-modelos que es adecuado para representar una solución dada. Los agentes utilizan los criterios idóneos para cada meta-modelo que representan. También se evalúa la compatibilidad de los grupos creados para resolver el problema de decidir que grupo final es la mejor solución.

Esta tesis se centra en proporcionar un proceso y un prototipo de herramienta para resolver el último paso considerando la incertidumbre que podría haber al solo conocer parcialmente un problema caracterizado y uno o más meta-modelos caracterizados. Por lo tanto el camino propuesto ha sido creado utilizando varios conceptos del meta-análisis de la inteligencia artificial de cooperación, la cognición Bayesiana, la incertidumbre, la probabilidad y estadística.

Résumé

Considérant un point de vue général de cette thèse aborde le problème de trouver, à partir d'un ensemble de blocs de construction, un sous-ensemble qui procure une solution à un problème donné. Ceci est fait en tenant compte de la compatibilité de chacun des blocs de construction par rapport au problème et l'aptitude d'interaction entre ces parties pour former ensemble une solution.

Dans la perspective notamment de la thèse sont les blocs de construction de méta-modèles et le problème donné est une description d'un problème peut être résolu en utilisant un logiciel et d'être résolu en utilisant un système multi-agents.

Le noyau de la proposition de thèse est un processus qui analyse un problème donné et puis il proposé une solution possible basée sur système multi-agents pour ce problème. Il peut également indiquer que le problème ne peut être résolu par ce paradigme.

Le processus adressée par la thèse consiste en les étapes principales suivantes: (1) A travers un processus de caractérisation on analyse la description du problème pour localiser le domaine de solutions, puis choisissez une liste de candidats des méta-modèles. (2) Les caractérisations de méta-modèles candidats sont prises, ils sont définis dans plusieurs domaines de la solution. On fait la chois parmi le domaine trouvé dans la étape précédant. (3) On crée un système multi-agents où chaque agent représente un candidat méta-modèle. Dans cette société les agents interagissent les uns avec les autres pour trouver un groupe de méta-modèles qui est adapté pour représenter une solution donnée. Les agents utilisent des critères appropriés pour chaque méta-modèle à représenter. Il évalue également la compatibilité des groupes créés pour résoudre le problème de décider le groupe final qui est la meilleure solution.

Cette thèse se concentre sur la fourniture d'un processus et un outil prototype pour résoudre plutôt la dernière étape de la liste. Par conséquent, le chemin proposé a été créé à l'aide de plusieurs concepts de la méta-analyse, l'intelligence artificielle de coopération, de la cognition bayésienne, incertitude, la probabilité et statistique.

Preface

Really knowing semantics is a prerequisite for anything to be called intelligence
-Barbara Partee

That's a notion of [scientific] success that's very novel. I don't know of anything like it in the history of science
-Noam Chomsky encouraging the machine learning researchers to understand the origins of human and animal nature behaviour instead of simulate it.

Today's software systems have an inherent tendency toward complexity. The popularity of distributed and complex systems - as those that run on mobile devices or sensor systems deployed in cities, forests and fields, or within the area of home automation - it requires to develop systems that go beyond traditional software development methods. The Multi-agent systems archetype is one of the paradigms used to fill such a gap. The MAS paradigm allows to model a system using schemes based on human social and organizational structure that currently solve complex problems everyday into human organizations and institutions. Many efforts have been made to simplify the development of MAS (Gómez et al., 2007). However, until now most part of these efforts have been done without coordination and towards different directions. This vast and diverse variety of choices makes difficult the initial choices for a system designer. In other words, the designer must choose among all the options and components an adequate methodology for the purpose of their system. This situation causes an initial uncertainty for any developer. Therefore it difficulty leads to the use of MAS as a viable option to develop a solution. In spite of this, we consider that we could improve such initial situation using a Multi-agent system to help making the initial decisions. Justifying such election phylosophically we recall the second law of thermodynamics (Kelvin, 1892), such a law prohibits two bodies at different temperatures can transmit heat from the cold body to hot body. The second law also is commonly expressed by stating:

> In an isolated system the entropy never decreases.

Such a law applies indirectly to initial uncertainty situation mentioned above. Initially the developer has a closed set of options and a problem to solve; therefore, he has to match the characteristics of his problem to components -belonging to one or more methodologies - that can bring a path to build a solution to such a problem. Thus, the level of experience and prior knowledge of existing methodologies take a key role in making initial decisions. Here we find several similarities with the Maxwell demon paradox (Kelvin, 1879): it proposes to employ a demon into an isolated system in order to reduce the thermic entropy. Thus, the demon activity consists in arranging the hot and cold molecules to reduce the thermic entropy within the closed system. A proposition to solve such demon paradox is done by recognizing the theory of information and therefore such theory justifies the energy needed to identify the hot and cold molecules. In other words the demon need knowledge to infer which molecule is hot or cold and put them in order. We propose to add agents similar as the demon - with experience and knowledge already acquired - in a closed system of options - solution components instead of molecules - with a high level of uncertainty because we don't know which components are the most suitable to solve a problem - considering the problem as the task of order molecules to decrease the thermic entropy -. So, the agents can help to find some possible solutions within the closed space of options.

It is possible from our point of view to make profit of this situation by using an approach to abstract information from existing methodologies and approaches automatically - invest in the experience acquisition - thus use such information to choose the most accurate components - match problem characteristics with components features - and therefore create a MAS where each agent acts as a component representative and look for other agents -other components - to create an optimal solution group. Therefore the final goal of our approach is to simplify the initial MAS conception process and thus to encourage the use of MAS.

Contents

List of Figures

List of Tables

Part I

Introduction

Introduction

The present work has been initiated by an already long cooperation between the laboratory LCIS - (Laboratoire de Conception et Integration de Systèmes) in Valence, France and CINVESTAV Guadalajara Mexico about Multi-agent specification methodologies. This work is under the direction and supervision of Pr. Michel OCCELLO, leader of the Complex Systems (COSY) research group and deputy director of the laboratory LCIS in Valence, France also by the Pr. Félix Francisco RAMOS-CORCHADO leader of the distributed systems research group in CINVESTAV Guadalajara Mexico. The present thesis work aims at developing an automated meta-analysis process for multi-agent software engineering. This thesis' work is considered as a base for the METALISM project that has been granted with the ECOS-ANUIES research project fund number M10-M02. This project also aims at simplifying the use of multi-agent systems and disseminate their utilization in the software development industry as a mature approach to create complex systems solutions.

Project Motivations

Existing methods are generally based on "problem to be solved into a domain" driven engineering, in other words, a given problem to be solved within a specific domain in which the methods are oriented. The fact is that we don't know if the domain method which is intended to solve the problem is the right one. This is an issue that must be carefully taken. An inappropriate domain of solution choice can make the MAS employed as not viable or inefficient. The need to deploy software to solve problems increasingly complex is a daily issue. The use of methodologies for the software design and layout is a fundamental requirement to reach a mature and successful software development. The multi-agent systems-oriented approaches represent an alternative development solution for complex nature problems, however, given the quantity of existing multi-agent approaches, the diversity of these methods makes tricky to choose the right approach, especially for inexperienced designers and for those not specialized in the area. The domain covered by each method is limited. In the same way, choose the appropriate domain solution to solve the given problem becomes an intricate task. In other words the designer must well know his problem and accurately identify their belonging domain solution. For example, methodologies such as ADELFE (Bernon et al., 2005) are oriented only to the realm of adaptive multi-agent systems, just as Gaia (Wooldridge et al., 2000) is directed only to a closed domain with static characteristics. So, the system designer must know

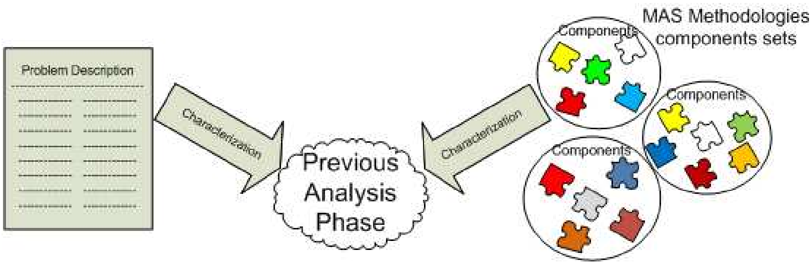


Figure 1: The previous analysis phase proposed in this work comes from two different sides of a same problem: the problem description characterization and the components o models characterization.

both sides of the way before choosing the route.

Based on the above facts we perceive that it lacks a supportive alternative capable to unify both parts. Such alternative must support the choosing task of the adequate MAS components specification, that will enable to build the target application, considering the problem domain.

We propose to cover this lack using a mechanism to guide the designer in the methodology choosing process suggesting the most suitable models and the best architectural integration method for his solution purposes. The creation of such mechanism would further make more achievable the use of methodologies based on multi-agent systems for developers, as well as would facilitate the extensive use of multi-agent systems, further expanding the options for conceiving and designing software.

Project Problem Description

The existing diversity of multi-agent methods that are oriented to an engineering of given problems in specific domains makes difficult the choice of the right approach. This means that when a problem is taken it is tried to be solved using any multi-agent method. This is usually done without previous analysis if the problem nature matches with the domain of solution covered by the method chosen. Thus, our main thesis problem is that there are many methodologies, many problems and many possible solutions for MAS. Therefore the decision to solve a problem using a MAS-based solution is not simple. Also considering that we have a low MAS development experience this becomes even more complicated and the probability of failure using MAS becomes larger.

The problem addressed in this project is the need of an overall preliminary analysis mechanism capable of unifying the models specification choices depending on the problem domain and the target application. This work focuses in one main problem related to the need of a previous step of analysis before to choose a multi-agent methodology to deploy a system. We can explain this main problem from two different sources (see figure 1). The first source is the problem characterization that is just situated in the problem description analysis. The second source is located in the components or models used to create a multi-agent system.

Thinking in an unified global software engineering approach to conceive multi-agent systems is not possible because domains and problems are very heterogeneous. However considering the kind problems which commonly are efficiently solved using multi-agent systems are:

- Simulation
- Troubleshooting
- Integration of Software and control system or systems.

We consider that a simple model for create MAS for all of them is not possible. Usually these problems are solved using approaches dedicated and specialized. For example how can we unify the design of applications as distant such as Internet Web Services, or Manufacturing Systems?

The multi-agent paradigm inherits features from various theories of decision and social. Each agent entity can be formed by many types of capabilities from a single reactive to a highly cognitive. Such entities are integrated in different architectures operationalizing models of agents, skills of interaction, perception and processing tasks within the environment.

From a social point of view multi-agent features are integrated across various modes of interaction and organizational structures related to the nature of the agents. This project focuses over the non generalizable model problem too, proposing a solution in the characterization of meta-models in the same previous analysis phase.

Thesis' Problem description

Our main problem is that there are many methodologies, many problems and many possible solutions for MAS. Therefore the decision to solve a problem using a MAS-based solution is not simple. Also considering that we have a poor MAS development experience this becomes even more complicated and the probability of failure using MAS becomes larger. We do not classify problems and we do not generate an uniform methodology, we are aware that there are several existing methodologies that are good for solving certain types of problems. Therefore our aim is to investigate which characteristics of problems tied to existing methodologies to generate a suggestion to the engineer who seeks to develop a solution based on MAS.

Thus the contribution of this thesis is limited to the creation of a process to match the characteristics of components abstracted from existing methodologies and domain with the application specification characteristics to propose a component group selection that represent a MAS-based solution for the application specification.

MAS conception common first steps

When a system designer starts to create a MAS he must consider a MAS methodology. Commonly each methodology provides processes, components and tools to develop the desired MAS. Nevertheless given the large number of methodologies, choosing the most suitable methodology becomes a sightless selection. Particularly for a beginner that is unfamiliar with the MAS methodologies

components provided and with poor MAS background. However such selection is crucial to deploy a MAS-based solution.
Moreover an inherent problem of the MAS methodologies is the different-methodology-components usability. That's why we cannot take components from different methodologies to create MAS-based solutions easily. However there are efforts as the method fragments approach where different pieces of the process are taken from different methodologies (Cossentino et al., 2011), nevertheless, it needs to adapt such pieces using a method fragment standard to make them usable.

The reason is that commonly the methodologies are not compatible between them, thus, their components cannot work together without a hard adaptation work. Lastly, it becomes hard to think in re-usability of already created solutions because they are specific solutions for specific problems. For example, an agent that runs into a MAS-based solution could not simply be employed for a different but related MAS-based solution. This is because the agent implementation depends on the specific interactions, environment and organization created for the specific MAS-based solution. Therefore adapting the already implemented agent to a brand new MAS-based solution could be a hardest work than starting a new agent implementation.
Thus, the issues giving origin to the creation of our approach are:

- When choosing the methodology that matches accurately with the desired application the system designer must have a good MAS concepts background and a good knowledge of the desired application domain.

- The designer must consider the difficulties of the methodologies dependency of components, for instance, a solution that needs components from different methodologies.

- Therefore, the quality of the system designer knowledge and experience influences the qualities of the MAS development process and the final product.

That's why we perceive that these development steps are directly linked with the decision-making field. To cover these issues, we propose a contribution that focuses on a preliminary phase to guide the system designer supporting their decisions in the first steps creating an approach that covers:

1. MAS Meta-modelization:

    - Decomposition and reutilization: The approach must decompose the existing and future methodologies components into meta-models to make them independent and reusable.
    - Meta-model Characterization Knowledge Base: The approach must use a characterization of each meta-model identifying and abstracting its features.
    - Knowledge or Experience acquisition: The approach must use statistical values to locate each meta-model feature into one or more solution domains. Thus consider such features as reliable or not within each domain according to the experience (stored as statistical values). These values are updated considering the system designer feedback.

- Meta-models constraints: The approach must consider compatibility issues between each meta-model to avoid the use of incompatible sets of meta-models. Also it must propose wrapper solutions when possible.

2. Application Specification as problem description:

   - Application Specification Input: The approach must receive as user input a textual or semi-structured document containing the problem description. Such problem description must be written within a software engineering context and declaring which standard or protocol (as UML for instance) has been employed.
   - Problem Characterization Knowledge Base. The approach must use problem characteristics abstractions.
   - Knowledge or Experience acquisition: The approach must use statistical values to locate each problem characteristic one to one or more solution domains. In other words each problem characteristic has a relation with one or more domains of solution, thus, such relation has a statistical value that defines if a problem characteristic is related or not within a domain of solution.
   - Application Specification Characterization process. The approach must receive the user input and characterize it using their Knowledge bases to identify their problem characteristics and thus the domain or domains of solution.

3. Matching Application Specification and Meta-models:

   - Matching process. The approach must match the problem characteristics and domain with the meta-models' characterizations and induce the most reliable meta-models to built a solution.
   - Characteristics Meta-analysis: The approach need a meta-analysis process to solve the decision making issues within the matching process.

Objectives

The most important contribution of the entire project approach is the decision support for the MAS developers, however, the present thesis focuses on the last item listed above. Therefore the major contribution of this thesis is the definition of a new meta-analysis approach performed by cognitive agents within a MAS that enables make decisions employing problem characteristics and meta-model features as statistical values. Furthermore each cognitive agent owns an internal module to perform the meta-analysis built upon a Bayesian cognition algorithm.

The main objective of this project is to reduce the difficulty and risk reduction using a multi-agent system as basis for a problem solution. Make the first steps of the development process simpler, easier and safer for the developers and system designers. In consequence make the multi-agent system approach more acceptable for the industry and spread its use.

In order to clarify the agents' decision making process and before introducing our contribution we must define the meta-analysis term as we understand it: the meta-analysis gathers statistical results from related studies. Within the medical field, a domain specialist usually choses such related studies (Sutton et al., 2000) from databases as Medline and PubMed (NIH, 2010) - where such studies are commonly stored - to make decisions using the experience gathered through all the studies. This process enables using all the previous analysis performed in these studies to make decisions for a current case. For instance, the meta-analysis could suggest using a specific medicine or treatment to treat an illness. Hence, to take advantage of the successful meta-analysis decision-making feature in the present work we propose to make use of a meta-analysis based process. Every aspect will be explained deeply in the section 3.

Proposed Approach

In general words this approach proposes to understand the problem from their characteristics finding hints to situate it into a domain. Within an analysis mechanism that takes into account problems features and their related domains. This mechanism must be situated before of choosing a multi-agent method. After this step we can situate our problem in their domain before start, but it remains to know which of the methodologies provides us with the right one domain of solution. To solve this adjacent issue it is necessary to characterize each component or model utilized in the construction of Multi-Agent Systems. At this level this work proposes to use a Model-Driven Engineering approach based in the Multi-Agent System vowels fragmentation. Characterizing meta-models to match with the problem characterization instead of utilize directly an existing methodology.

Motivations of our approach

Often we have a problem to solve and many parts to build the solution. Choosing the right parts to build the solution to a given problem requires knowledge experience about the utilization of such parts within certain domain and circumstances related to the problem to solve. The acquisition of this experience has a cost determined by the information theory. This experience allows us to distinguish between all the possible parts. Thus we can discriminate which parts are useful and which not. Such experience is similar to one of the solutions proposed to the Maxwell's demon paradox (Kelvin, 1879) (ordering the hot and cold molecules). So, in such context the experience information is a valuable resource for building a solution. We propose an approach to manage such experience employing the multi-agent system (MAS) paradigm to combine individual knowledge experiences (obtained by independent specialized entities) to solve new problems partially unknown (uncertain). Our approach must be seen from two point of views: (1) General case, where we consider it as a general problem that rises from several different domain situations considering

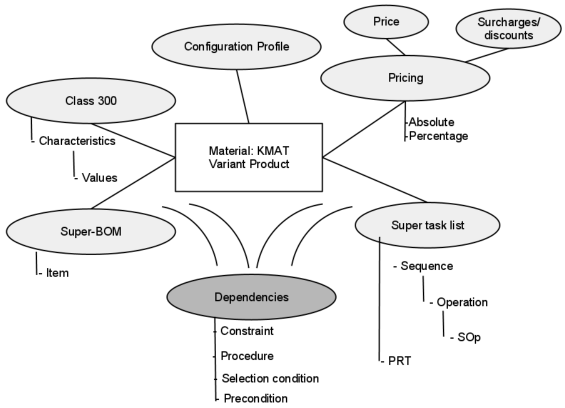


Figure 2: Diagram of a Variant Product KMAT from SAP

a problem to be solved and solution entities to build or compose a solution for such a problem and (2) Particular case, that belongs for our specific case using a software engineering application specification as problem to be solved and MAS meta-models as solution entities.

General cases

Considering a general case of the proposed approach it can be also employed to solve complex problems related to the sensor networks, fleets of robots, drones and virtual agents to build entities composed solutions or solution process with a focus on collective work and cooperation.

From other point of view the approach can be considered as a general case of configuration of material where several components within a catalog could be employed together (considering their inter-compatibilities) to build several different products within a production industry. Similar as it is performed in the KMAT (see fig. 2) within the configuration process of the logistic and variant configuration module of SAP (Blumohr et al., 2011). Such a configuration process allows to make efficient the enterprises production performances because it makes able to automate and test the possible configurations of materials and reduces the complexity and cost of these tasks. In our case we propose a similar path but we have an application specification instead of a KMAT, a Software Components instead of items and we provide with the meta-models something similar as the task list that helps to describe how to assemble or produce the KMAT in order to deliver a product to sell.

Nowadays the complex systems development is a rising trend in the software engineering field. This is directly linked with the growing demand of mobile systems, sensor networks, home automation systems, extended reality and virtual simulators etc. The foundation of a new generation of paradigms is considered as an important tendency. These paradigms will be responsible to facilitate the complex system design and development in the next years considering the 21st century system requirements (Joann Roskoski, 2003).

The multi-agent paradigm is based on the social interactions from the human and nature real life.

Figure 3: NASA In-situ resource utilization logo

It rises as a pertinent archetype to analyze complex problems and conceive solutions for complex systems (Michael, 2002). Nevertheless the development of MAS-based solutions has various issues that make the industry perceive the MAS solutions as expensive and fail prone (Dastani et al., 2004). We state that the main issues are related to the decision-making field, thus in this thesis we propose an approach to support such decisions. Such an approach comprises a multi-agent system (MAS) whose agents use a Bayesian cognition algorithm in order to perform meta-analysis of data to make decisions.

Currently for process performed by a group of individuals, such as a fleet of robots, is common to employ individual behaviors of ants, bees and fireflies that allows the emergence of organizational behaviors (Beekman et al., 2008). These approaches are accepted and reproduced today to solve optimization problems and create communication routes. However such solutions are second when it requires coordination of several independent entities such as the use of intelligent sensor networks in automatic car driving, fleet of robots and unmanned spacecraft. In the handle of dangerous situations such as emergency nuclear plants or underwater exploration and space in situ resource utilization.

The last one represents one of the main trends towards the deep space exploration:

- Energy Space Race. Considering the perspective of energy generation based on nuclear fusion employing Helium-3[1] (Troyan, 2009). In such a space race there is involved several countries: Russia (Roscosmos), India, China and United States of America (NASA).
- In-situ Resources Utilization. NASA has an special research group NASA ISRU (NASA, 2011) (See fig. 3) that aims at developing new robotic technology in order to collect and transform

[1] A radioactive isotope of Helium that is rare on earth but common in space, specifically on the earth's Moon and Jupiter

resources in the deep space to use them has energy sources, materials for rockets or for life keeping systems.

- Deep Space Exploration. Roscosmos is re-taking the new Russian space automated labs and probes development as the Phobos-Grunt (Roscosmos, 2011) probe has been created in order to explore the Mars' moon Phobos returning around 50 grams of material from their soil (See fig. 4). Nevertheless the Phobos-grunt probe has been doomed in the near earth orbit and failed to reach the route to the Mar's moon. The first failure investigation shows that the software system has failed because a heavy charged particle from outer space has provoked a system failure and the operative systems has been pushed into a low consumption mode (ITAR-TASS, 2012). It displayed that the main computer has been overloaded and thus blocked all the possible earth communications and remote reprogramming of the probe. In conclusion the failure seems to be part of a poor quality electronics and a centralized paradigm. Maybe an agent-based and decentralized schema could be able to help in self recovering of the failure and save the probe instead of lose it.

- Long term Space Exploration missions. The ExoMars (Agency, 2012) is a ESA - NASA joint Mars robotic exploration project of several years that originally was designed to be multi cooperative rover capable. It means that at least two rovers would be involved in the exploration of the same place. The rovers must be cooperating between them in order to perform experiments and make measurements. And also they will be able to gather materials and elements in order to await for a future robotic mission that will take back the gathered elements to earth. In conclusion the project proposed the make cooperate several robots with different skills to achieve different complex tasks. Such a project is a plan that is not yet achieved however it states a complex cooperation between robots in a deep space exploration mission.

Therefore the development of the required technology to gather such space resources is increasing their activity and cooperation approaches to automate process as the proposed in this thesis that are well-related to provide automatic cooperation capabilities to the existing space rovers and robots (specialized entities) in order to exploit on space in situ resources (complex problem to solve).

Today's systems have evolved in a way that demand a capacity for interaction between them as individual intelligent systems that collaborate between them. Among the proposed linking solutions are the middleware based, web-service composition, cloud computing or grid computing. However, the organization of entities and their interactions allows us to build systems inspired in the human nature to solve problems collectively, using individual specialist skills (possessed by every human).

The paradigm of multi-agent systems stems from the interactions within a society or community which allow several agents with different expertises to perform different tasks but with a common collective process.

This approach proposes to exploit the experience acquired by a single specialized entity (sen-

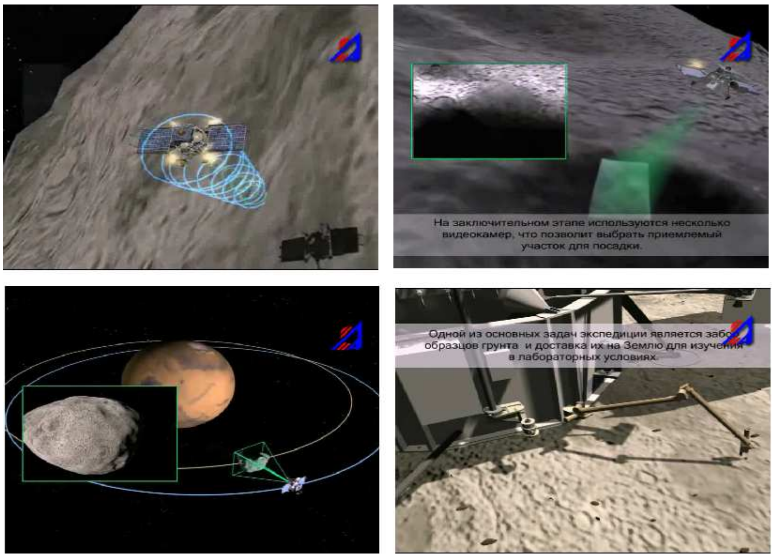


Figure 4: Phobos-Grunt unmanned mission to mars simulation, the images show how the automated probe characterizes the Phobos soil in order to land and take some soil samples, there is also a system to analyze the soil samples that needs an automated cooperation between a robotic harm, a sample analyzer within the probe and the sample return capsule to earth.

sor, robot, agent, meta-model components, OSGi bundled packages, etc.) performing a given task (participation in a solution or successful settlement process to resolve a problem or a fraction of a given problem) within a group of entities that interact and self-organize to construct new solutions (choose appropriate entities) for new problems identified.

Where a relationship "performed task"(solution) - "problem characteristic"(problem) can be built. And so each entity is considered as a piece of the solution and in turn creates an efficiency ratio of such entity with one or more characteristics of the problem to solve.

Finally, it results in a set of entities that represent the solution group proposed for a given problem.

Fundamentals

Identification

A collective task requires the participation and consensus of all participants, including the decision-making. For example, individually a robot can make decisions based on readings from its sensors. This way a robot can learn from its environment and adapt to it developing their individual specialized task (exploration, excavation, collection, moving objects, etc.)..But when we have a fleet of robots. Each robot learns and adapts to its environment on an individual basis according to their special abilities to do their specific job. When we want that the fleet of robots perform a collective process that requires the combination of their individual abilities and organization requires consensus to achieve the collective goal. It is here that an approach based on behavior of ants or bees may be insufficient for the organization of more complex tasks. Especially if the problem to be solved is unknown is therefore to solve something in a high uncertainty about how to proceed.

An example of this type of situation is shown in Figure 5. Showing a fleet of robots with individual capacities where:

- The first case involved the extraction of minerals on asteroids in space.Robots must analyze an asteroid at time to explode it determining the order of performance of each of them and adapt to unplanned situations. The major uncertainty of the mission are asteroids with varying chemical compositions and diverse geological forms making it difficult to create a general organization and interaction process.

- The second case involved the repair of a disaster at a nuclear power plant. The robots must deal with a hazardous environment with high uncertainty about how to proceed with repairs. The robots must detect faults, leaks of radioactive water and cracks in reactor. Then, create plans to repair such damages. It should be noted that such tasks (including the proper order to make repairs) are performed as a human being do.

In software engineering existing approaches propose the reuse of components used to create solutions for different types of problems, and therefore are efficient individually to solve certain characteristics of a given problem within a solution domain:

- Meta-models describing fragments descriptions of solution to build systems that solve problems that can be solved with software.

- OSGi type components that are dynamically loaded and can work together to create software solutions to certain specific problems.

- Building blocks that are defined as parts to create software solutions, similar as meta-models.

For this component approach are required expert designers who know about the efficiency of each component to specific problems and their compatibility with each other to build solutions. This thinking leads to an automated approach using each component as an entity with specific skills to solve certain parts or features of a problem. Which would implement a scheme of self-organization of software components that automatically build solutions based on given problems. This uses the existing experience about the efficiency of each software component to solve a certain type of problem under a given domain and the compatibility of other software components interaction (or restrictions).

Problem identified

This exposes the lack of a collective intelligence approach that allows individual entities to exploit capabilities dynamically combined and organized. In other words, the obvious problem is the lack of an approach to designing a capacity similar to human society to self-organize and create solutions

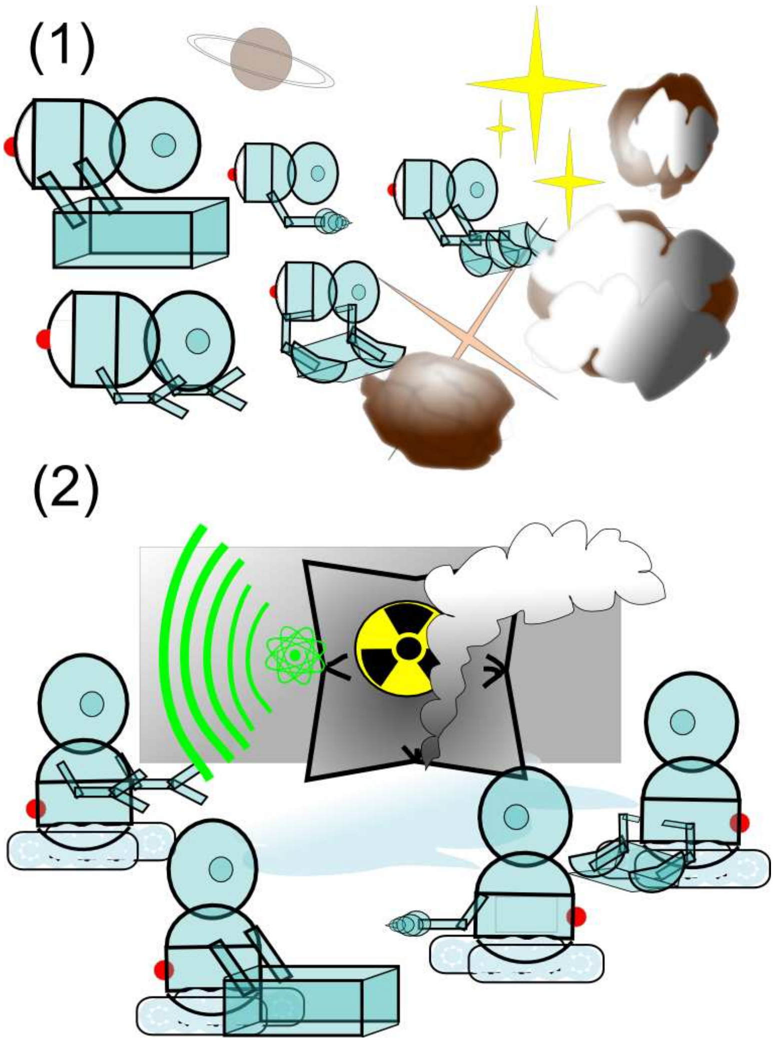


Figure 5: Situations that motivate the proposed approach is illustrated where a fleet of robots with individual specialties that (1) must collect minerals from the asteroid belt and (2) should perform repairs in an environment of high radioactivity and dangerous to humans.

to new problems using the characteristic of the existing expertise of each individual within a group organized.

Make an approach with the features described can lead to complexity problems and some solutions for decision-making capacity of artificial intelligence are considered AI-Complete (NP problems like) in addition distributed and independent nature of the entities in question makes a centralized approach a not viable option. However, in the next sections we will discuss the proposed strategies to make this approach viable in despite of the AI complexity problems.

The Society of Decisions

Today there are approaches that conceive of teamwork or group such as the swarm of robots or proposing a form of collective intelligence as swarm intelligence is (Trianni et al., 2008). On the other side the field of artificial intelligence, there are approaches as artificial evolution, logic, probability and statistics (Russel and Norvig, 2003) that work in an individual or specialized manner, such approaches are found in agents, robots and sensor systems individually (Bessière et al., 2008).

Usually there are approaches that demonstrate success within the field of swarm intelligence. Usually inspired by the collective behavior emerging from interactions of individual behavior in nature and finding the shortest path using the behavior of the ants following the path of pheromones, the order of formation of the flock of birds, communication light of fireflies, and so on.

There are approaches that combine the groups based on these natural approaches to artificial intelligence individual for example (Trianni et al., 2008) proposes the use of swarms of robots based on the individual behavior of each individual in the group but uses artificial evolution to learning and individual behavior . Thus it does that through simple behaviors of each robot to perform more complex collective tasks.

In (Minsky, 1986) a natural approach is proposed based on human brain function like a society of agents. Similarly we propose to use agents as a society that receives information from its environment and makes individual decisions using individual experience and then through interactions take a collective decision. Thus functioning as collective intelligence. Minsky's society of mind work also proposes two types of agents, some specialists in solving problems and others in the choice of these specialists, management and planning for troubleshooting. In contrast we propose to use agents that perform two functions: (1) represent an individual Specialist (robot, component meta-model, test software, etc.). And (2) planning the efficient use of each them together and in a certain order.

Particular Case

Our approach proposes the creation of a distributed agent society (at least in abstract form) that allows solving a given problem in emerging way. Such problem solution arise combining the entities that are specialized in certain task (like building blocks) that together and in a certain order are capable of solving complex tasks. To achieve our proposal we have two important starting points: (1) the problem or complex task to solve (2) the specialized entities or building blocks. To then use a SMA-based architecture that allows to model the interactions necessary for the development of collective artificial intelligence.

Complex problem or task to solve

Considering the first part we have a problem coming and for which a solution must be built. Considering the approach "divide and conquer" we propose to divide the problem characteristics or patterns of problem. Where each characteristic is situated in one or more domains. For this, the features must be identified and therefore need to be stored in a knowledge base with values that determine the effectiveness of each feature within each domain. Then when we find a new problem to solve we should take as analysis basis the existing characteristics stored in the knowledge base. The following is a process of identifying features to further assess the global domain trend of the features found. In other words, find the domain or domains where the performance values are higher. This process get important clues that later help to reduce the number of candidatures of specialized entities.

The acquisition of problem characteristics can be done through a learning classifier system that allows the identification and dynamic update features. The learning process then involves the regulation of efficiency values within each domain. To start it can be done from data gathered from a query answered by human specialists (software experts for instance). Or also gathered through an automatic process (using artificial evolution, genetic algorithms, etc.). Finally it can be gathered on-line from the feedback results.

Thus we propose to use a scheme based on probability and statistics that can regulate the system settings as experience is gained. It also allows a dynamic update of knowledge bases and values of efficiency.

Specialized Entities

Similarly as a problem is characterized, it is necessary to identify the features of each specialized entity. At the same time placing each entity property within one or more domains with values that determine their effectiveness within each domain. Unlike obtaining the characteristics of problem in the case of the specialized entities they can be defined based on features, that is, they can start with a predefined set of features and defining a new specialized entity that uses a subset of them. Therefore there is a knowledge base of specialized entity's features. Moreover a knowledge base of available or known specialized entities features.

The acquisition of features lends itself to be direct, for instance, written by a designer of specialized entities. For instance an expert software designer that defines reusable meta-models. However, it is possible to take an automated approach, as in the case of existing specialized entities, for example existing software components characterized automatically as specialized entities. The learning process as in the case of the characteristics of problem must be on-line. We can employ initial values based on a survey-based gathered data from expert human entities designers.

Thesis Organization

The present thesis is organized in three main chapters, the first is related to the state of the art and general context of relevant topics, the second relates the principal contribution of this thesis that is the basis for the project, and the final chapter shows the final evaluation of this work.

Each part is divided in several sections:

In the first part:

- This introduction is located as initial chapter.

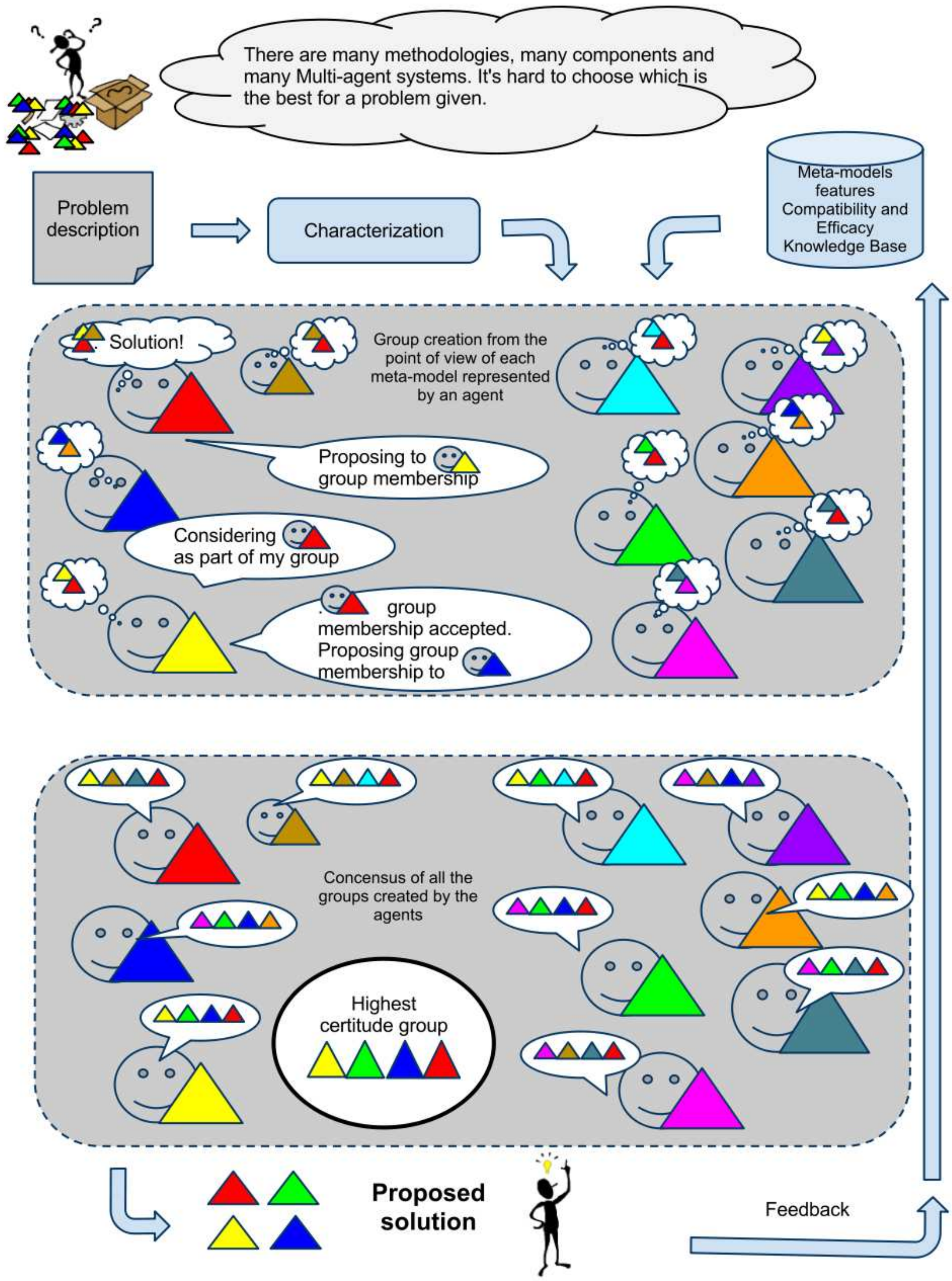


Figure 6: The proposed approach main process overview.

In the second part part:

- Chapter 1 is composed by brief reviews, state of the art, introduction and general context of different related topics as complex systems multi-agent systems, emergence, collective intelligence, self-organization and discussions about the MAS based solutions and the emergent collective behavior within MAS.

- Chapter 2 talks about software engineering covering the MAS Decomposition, the model driven engineering (MDE), the MAS methodologies context and finally discusses the difficulties of a designer when he make a choice of MAS methodology.

- Chapter 3 introduces the term Meta-analysis and presents briefly the path that it follows, the biases and source of troubles, the cases and common methods employed within it and finalizes discussing the relation of it with the present thesis approach. It also discusses the decision making artificial intelligence related issues. It introduces the terms decision-making, Bayesian cognition and how does it work as artificial intelligence employing probabilistic reasoning. It makes a brief introduction about statistics, probabilities and Bayesian cognition context employed in this thesis document.

In the third part part:

- Chapter 4 discloses the approach main overview covering the contributions, activities and phases.

- Chapter 5 introduces the preliminary ideas and fundamentals of this thesis approach, first in a general manner and then it discusses the particular problem of our thesis and their goals. It also introduces the satellite phases of the approach

- Chapter 6 proposes the solution architecture and describes the main contribution of this thesis: the matching phase. Along this chapter is described how the matching engine is built, the internal MAS architecture, the artificial intelligence employed by the agents and how does it performs the collective task. Finally it discusses the overall contribution.

In the fourth part:

- Chapter 7 shows the tool specifications disclosing the diagrams and use cases of the prototype developed to test the approach and how does it work to help the designer to make decisions. It also displays the GUI of the prototype.

- Chapter 8 shows a case study and shows the results obtained following each phase using the present approach and finally compares the outcome with other obtained through common MAS human development. It also discloses the application time and resources profiling measurements of the prototype.

- Chapter 9 it remarks the overall conclusions and describes the future work and possible extensions for this approach.

The fifth part comprises:

- Appendix A introduces fragments of the XML Knowledge bases file examples employed in the prototype tool. It also shows a fragment of the Ontology OWL file.

Part II

State of the art and Context

Chapter 1

Multi-agent systems

1.1 Complex systems

In order to define a complex system we need to know first what is a system. The term system comes from latin "systema", and also from Greek "systemat", from the term "synistanai" combined from "syn-" + "histanai" that means "to cause to stand". According to Merriam-Webster dictionary (Merriam-Webster, 1981) a system is defined as:

> ... a complex unity formed of many often diverse parts subject to a common plan or serving a common purpose.

Therefore a system is considered a set of interacting or interdependent components forming an integrated whole. In computer science, a system commonly is related to a software systems built upon a structure of components that has a communication process between such components.

The term complex comes from Latin "complexus" that means totality and embrace. As stated in (Corning, 1998) a system is complex when it has the following attributes:

1. Composition. When a complex phenomenon is composed by many parts, items, units, or individuals.

2. Interactivity. When there are many relationships or are performed many interactions among such parts.

3. Emergence. When these parts produce combined effects or synergies that are not easily predicted and may often be novel, unexpected, even surprising.

In (Joslyn and Rocha, 2000) a complex system is also considered as a system composed of parts that are interconnected. Such system at large displays one or more properties not obvious from the properties of the individual parts. Commonly an entire system behavior among the possible properties not evident at individual level.

There are also several kinds of complex systems:

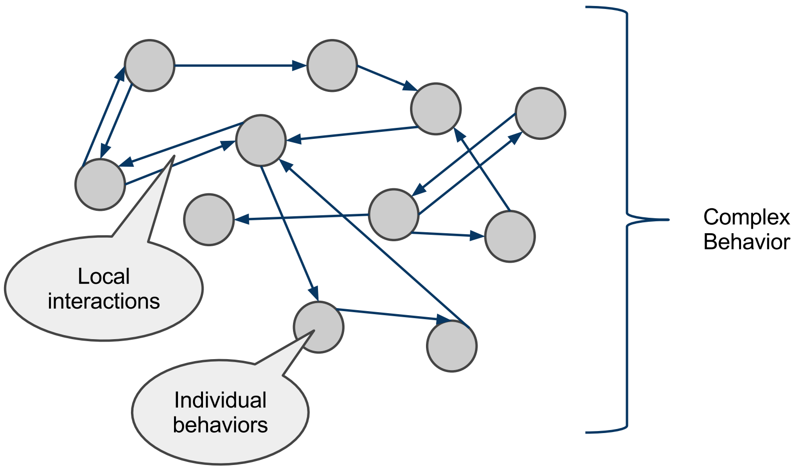


Figure 1.1: Complex Behavior emerging from a complex system.

- Chaotic. According to (Hasselblatt and Katok, 2003) such systems are sensitive to initial conditions. It means that each perturbations at beginning will make different the future system behavior.
- Adaptive. It is commonly considered as a special complex systems case they are composed by interconnected elements and with skills of gathering experience and change in order to adapt to the environment. For instance of adaptive complex systems we can list, the human brain, the ecosystem, social systems, etc.
- Nonlinear. This kind of systems usually owns a behavior that is not subject of superposition. In other words its behaviors

A system is complex by the diversity and the multitude of interactions that it uses. Usually such systems are distinguished from others by the impossibility to identify all elements and understand the dynamically updated interactions (See fig. 1.1). This usually entails the absence of a total control and the irreversibility (any action cannot be reversed to change the dynamic to return with certainty to one of the preceding equilibrium states). Complex systems can be divided into different levels of interaction that enable to the simple elements to be added in more advanced components. These same components enable the emergence of well organized and hierarchical structures that interact strongly among themselves and with their environment. The structures that emerge then there cannot be understood simply from the entities utilized. Complex systems often are natural and shape the subject of active studies within domains such as physics, biology, human sciences and social and cognitive sciences. In computing systems, the systems of information, supervisory and problem solving are becoming more and more distributed, open, large scaled and heterogeneous. Their interconnections becomes so complex and crossed in such way that they exceed the overall understanding that a real human being can have doing complex artificial systems.

Therefore in this thesis we consider that a system become complex according to the quantity of entities and interactions need within it. Also considering the emerging collective behavior linked with such entities.

1.2 Introduction to MAS

In (Russel and Norvig, 2003) an agent is defined as an intelligent and autonomous entity that can perform a determined task. Therefore an agent could work in a single-agent environment, solving an individual task, like solving a puzzle; or in a multi-agent environment where there two or more agents performing individual task and group task through interactions.

According to (Weiss, 1999) there is no universally accepted agent definition. However, the autonomy skill is a features that is the most accepted into agent's definition.

In (Selten, 1975) an agent is defined as a computer system situated in some environment where it can perform autonomous actions (and also perceive changes) in order to meet its design objectives.

Wooldridge makes a difference between an agent and an intelligent agent (Michael, 2002):

- An agent: is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives
- Intelligent Agent: requires to be reactive, proactive and social.

However other definition for an intelligent agent employed in (Lind Padgham, 2004) declares an Intelligent Agent as a piece of software that is composed by the next features:

- Situated: existing within an environment (See fig.1.2).
- Autonomous: it is independent, therefore not controlled externally.
- Reactive: it responds to changes perceived in its environment.
- Proactive: consistently seeks achieve new objectives.
- Flexible: has many ways to accomplish their objectives.
- Robust: recovers from failures and unexpected situations.
- Social: interacting with other agents and the environment.

According to (Wooldridge, 2009) a multi-agent system is defined as a system that is composed by numerous intelligent agents that interact together to perform a task together. Also it states that we must consider different point of views to define the multi-agent paradigm:

Figure 1.2: Agent interacting autonomously with their environment.

- Software Engineering. The multi-agent paradigm is considered as an approach that enables to design autonomous software that can interact with different parts within a distributed system (See fig. 1.3). Usually the interaction capability it is considered as the most important feature provided by this approach. Moreover the multi-agent paradigm enables to design architectures to create software solutions to complex artificial systems.

- Distributed and Ubiquity. Considering the distributed systems approach the multi-agent paradigm provides an approach from which one can conceive distributed solutions employing agents. The ubiquitous point of view considers that the multi-agent paradigm

- Human societies tool. From the perspective of human societies, the multi-agent paradigm abstracts their individual specialization, organization and interactions features. It enables also to design a human based society simulation.

The term "multi-agent" appears as an innovative and effective paradigm to artificial complex systems modeling. The multi-agent approach inherited from various biological, social and decision theories. Multi-agent entities can be formed in many kinds of capabilities from the most reactive to the most cognitive. Such entities are integrated into different models operationalizing agents architectures, interaction and perception capabilities, and processing over the environment. From a social point of view multi-agent devices are composed of various models of interaction and organizational structures linked to the nature of the agents. The design of a system over the multi-agent concept is a recent approach in the software engineering field, however, for its realization numerous methods has been conceived (Gómez et al., 2007). They have been developed with the aim of being a basis for the software engineering development that enables a simplest way designing a multi-agent system. Existing methods generally are based on problem to be solved into a domain driven engineering, in other words, a given problem to be solved within a specific domain in which the methods are oriented. The fact is that we don't know if the domain method which is intended to solve the problem is right. This is an issue that must be carefully taken.

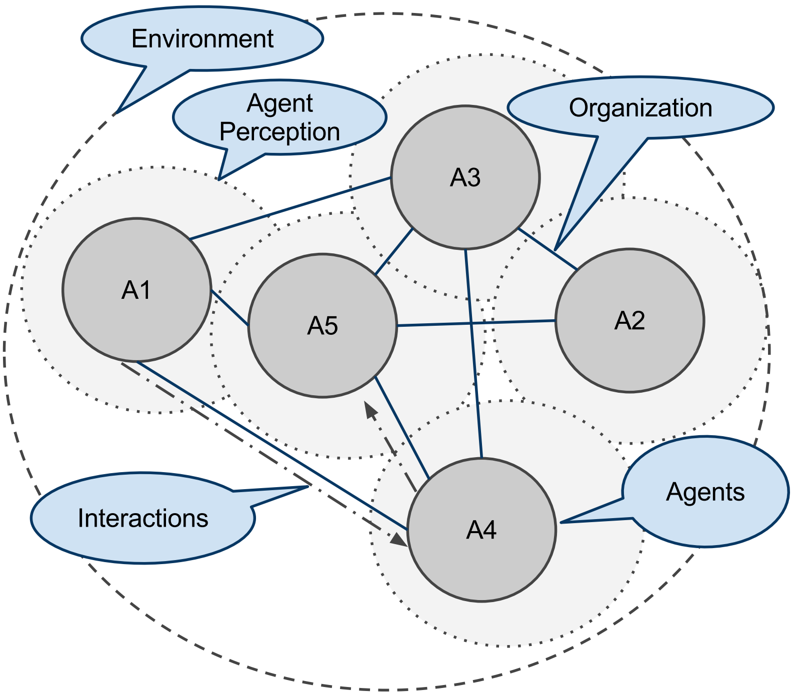


Figure 1.3: A general idea of a multi-agent system

1.2.1 MAS application areas

In order to identify the MAS Components features we have analyzed the literature and we have listed the following features that are employed for describe each MAS component:

Multi-agent systems are usually related as an appropriate approach to the following topics:

- On-line trading, (Rogers et al., 2007) Where the agents can learn and apply several bidding strategies to minimize cost and maximize gain. So, employing agents as traders could improve the income.
- Disaster response,(Schurr et al., 2005) In this case is commonly proposed that the agents are led by a human. They show important information about the place where the disaster occurred. However it is possible to employ agent organizational capabilities within robots to perform risks tasks.
- Modeling social structures (Sun and Naveh, 2004). It profits the human individual specialization and thus society structures abstracting such existing structures like institutions, human working groups, teams, etc. to create systems inspired in those structures. Also to simulate such social structures and evaluate situations within them.

However we list some of the main applicable areas of agents: (Lind Padgham, 2004)

- Remote control. The most significant example of remote controlling is described in the work of (Muscettola et al., 1998), where an agent based system performs the space exploration million of kilometers far away in space. It is performed through remote control from earth, however, considering the very long distance, there is a MAS-based system that must take crucial decisions when there are no communication.

- Substitution of humans. Some dangerous tasks like space or deep water exploration, anti-bomb or nuclear plants
- Human-like behavior. Human emotional simulations are performed using agents. For example as described in the work of (Ramos et al., 2005).
- Simulations, virtual drama, film-making. Several films like the trilogy of "The Lord of the rings" (New-Line-Cinema, 2001) or "The day after tomorrow" (20th Century-Fox, 2004) have employed MAS-based technology to recreate scenes performing massive battles in fantasy worlds. This is a believable virtual drama employed in commercial films.
- Intelligent assistants. There is a tendency to create MAS that work as assistants, for example for web-services composition (Abrougui et al., 2009) where a human user can request a set of agents to create a composite service based on such request. In (Maes, 1994) there is an agent that learns from the user interaction with an application and then imitates him to reduce the quantity of work and information overload. The actual state and the future of such intelligent assistants is addressed in (MIT, 2009).
- Electronic commerce. According to (Luck et al., 2005), the electronic commerce is one of the key fields of MAS. In (Rogers et al., 2007) is performed a study about the bidding strategies within E-bay auction system. Such strategies are considered to be employed within agents to perform such bids automatically and optimize the cost-benefit relationship.
- Manufacturing. The surveys about manufacturing systems in (Shen and Norrie, 1999) and (Qiao and Zhu, 2011) shows that the MAS-based solutions in manufacturing are increasing each time providing new ways to solve manufacturing efficacy problems.
- Business process management. In (Jennings et al., 1998) is proposed an agent based approach that shows how agent technology can improve efficiency by ensuring that business activities are better scheduled, executed, monitored, and coordinated.

1.2.2 MAS Components features

Depending on the type of agent, it comprises different features that are relevant to the actions performed by the agent, therefore, in the next lines we enlist the main agents properties according to (Michael, 2002):

- Being situated. An agent knows where is situated within an environment.
- Proactive. It must look to achieve goals.

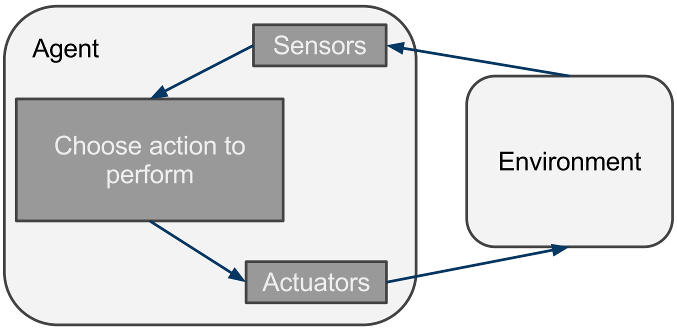


Figure 1.4: Agent interacting with environments using sensors and actuators to choose the next action to perform

- Reactive. It must react to external stimuli, coming from external agents or the environment.
- Social abilities. Skills like create group, perform a role within an organization, etc.
- Cognitive abilities. Analyze the current status and make decisions to perform a task depending on such status.
- Deductive reasoning. Such reasoning is commonly related with logic rules and properties to deduce something based in logic facts.
- Inductive reasoning In contrast with deductive reasoning the inductive one focus on mathematical induction moreover on probability and statistical facts.

There are also different kinds of environments, usually the agents are situated within an environment and perceive the changes through sensors and perform actions using actuators (See fig. 1.4). The environment is different according to their nature and it must be defined for every MAS as a task environment. We enlist the following list of properties based on the proposed by (Russel and Norvig, 2003) (See also fig. 1.5):

- Observable level. Its about the observability level within an environment. From the fully observable, when the agent sensors perceive all aspects relevant to the choice of action to perform, to partially observable, when only some aspects are gathered for example using only a type of sensor makes to perceive only a part of the environment.
- Deterministic-Stochastic-Strategic. The environment is deterministic when the current state has been determined by the previous state and the previous actions executed by the agents, for example when is a closed virtual environment and only the agent's actions modifies it. Is considered stochastic when its unpredictable, for example when the agents are sensing a physic environment: an explorer rover in a far away planet that must navigate in a unknown environment. An environment is strategic when it is deterministic except for the agent's actions.

- Episodic-Sequential. An episodic task environment is divided in episodes. Each episode consist of an atomic single action and usually each episode is independent from the last one. An episodic environment example is when an agent is arranging elements by a certain order, it evaluates each one at time independently from the others elements. In the sequential environments the current decisions affects the future decisions. For example when an agent is playing a board game its current movements will determine its future movements.

- Static-Dynamic. The environment is static if there are no changes when the agent is deliberating. For example playing a board game each player has a turn, so there are no changes while one agent is deliberating about a movement. On the other side a dynamic environment change while the agent is considering which action to perform, for example the rover exploring an alien planet driving to explore an area while the alien planet weather is changing. A semi-dynamic environment is in the middle of the static and dynamic, an example of such environment is a chess game where each movement is limited in time.

- Discrete-Continuous. This property describes the state of the environment, how the time is handled, the actions and perceptions of the agent. A discrete state is for example in a board game that has a finite number of different states, also a discrete way to perceive and act. A continuous state is like a rover camera that is continuously perceiving the environment state, and continuously acting according to that.

There are different Interaction properties described by (Occello, 2000). These main properties are described in the following list (See also fig. 1.6):

- Perception interaction. This is a form of perception that involves a direction from the environment to the agents, for example, a net of distributed sensors, where each sensor is owned by an agent (Jamont et al., 2010), the environment is measured continuously from the environment to the agents.

- Action interaction. This interaction occurs when the interaction comes from the agents to the environment, for example, an agent arranging objects modifies the status of the environment in such a way.

- Cognition interaction. This interaction is performed from agents to agents usually through messaging, for example, in a net of distributed sensors (Jamont et al., 2010) owned by agents where they communicate between them to pass messages.

- Dialog based. This interaction is derived from cognition interaction but employing human like dialogs, similar as proposed in (Reilly et al., 1996) where the agents interact between them using text-based messages to create dialogs. Also its possible to interact with a human user, in such case, the interaction becomes a mix of perception and action interactions.

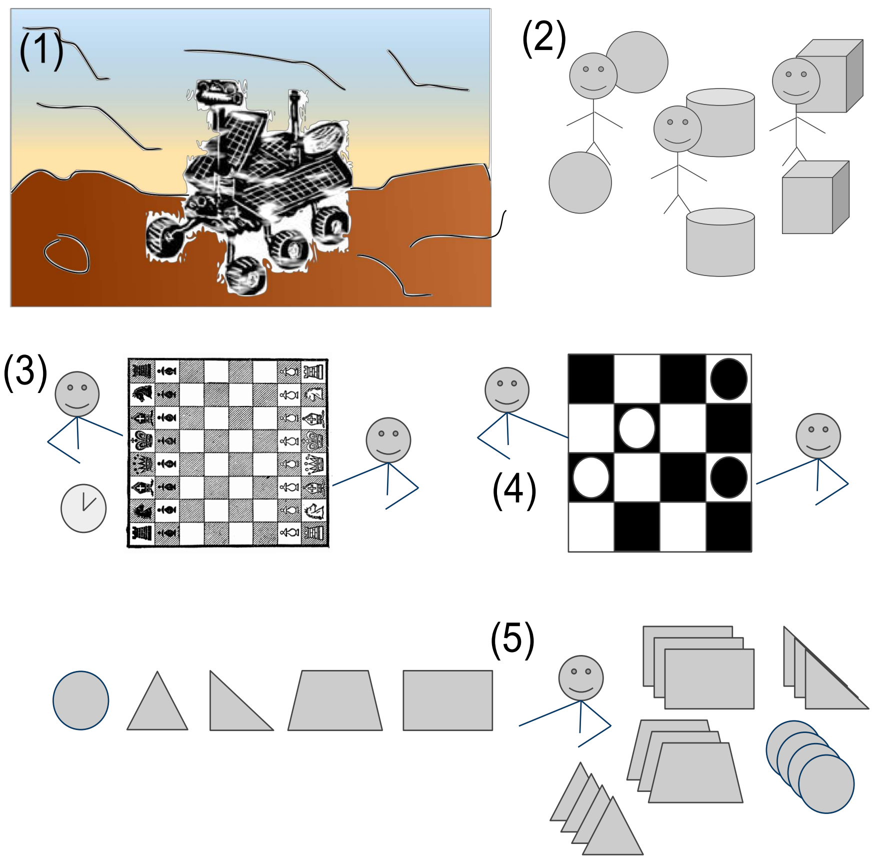


Figure 1.5: Some examples of environment features: (1) A rover exploring an alien planet (2) A closed virtual 3D environment, (3) A chess game with clock (4) A board game (5) An agent arranging elements

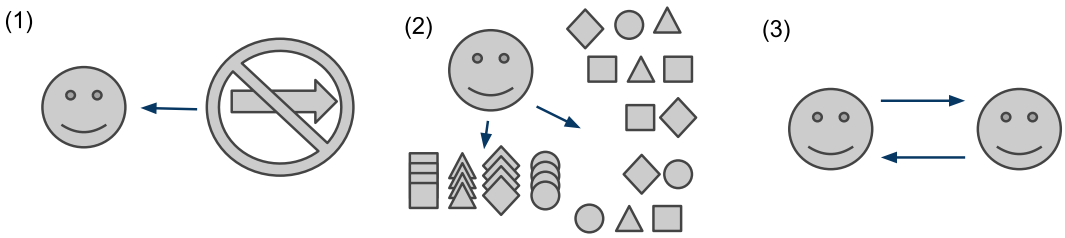


Figure 1.6: Some examples of interaction features: (1) From environment to agent (an agent considering a signal found in the environment) (2) from agent to environment (an agent arranging items of the environment), (3) Between agents (agents communicating)

According to (Baeijs and Demazeau, 1996) there are various organizational structures for MAS. In the study of (Mintzberg, 1979) there are three process of coordination that permit to identify three main families: groups, hierarchies and markets.

- Cooperative work. Two or more agents make accords to work together sharing a resource in order to achieve a common goal.
- Supervision-based. This is a hierarchical case when an agent supervise others in order to regulate their tasks. In other words when an agent leads a group of agents, for example, to develop a task the agent leader divide such task in subtasks to be performed for the subordinated agents.
- Standardization. When there is an agent that acts like an authority and place rules to be followed by the other agents. Such rules are applied to specific cases.

About the organization properties scope defined by (Baeijs, 1998) we have enlisted also the following (See fig. 1.7):

- Knowledge (an agent knows another). It is when each agent knows other agent both works independently for a collective goal.
- Client-server, communication (an agent communicates with another). When an agent asks another for a service not necessarily for the same goal.

1.3 Discussion: MAS based solution

As we understand a MAS based solution requires at least two agents, such agents will develop a role within an environment, such agents can develop interactions between them and thus organize them to work together. Considering the vowels approach (Demazeau, 1995) we can arrange different types of agents, environments, interactions and organizations. Nevertheless we must consider that not all of such types of components can work together. Therefore we must list a set of rules to make evident such compatibilities.

In order to match the vowels approach with the traditional software engineering approaches we have found in the model driven engineering (MDE) the best companion to apply such MAS decomposition approach. As we discuss in section 2.4 the MDE defines the basis for create meta-models. However we consider that a MAS based solution can be constructed from vowel components, in our case MAS vowel-based meta-models.

1.4 Collective Behavior and Emergence

The complex systems are usually considered solution archetypes for problems that cannot be solved using a centralized approach. Therefore such archetypes are based on distributed approaches that

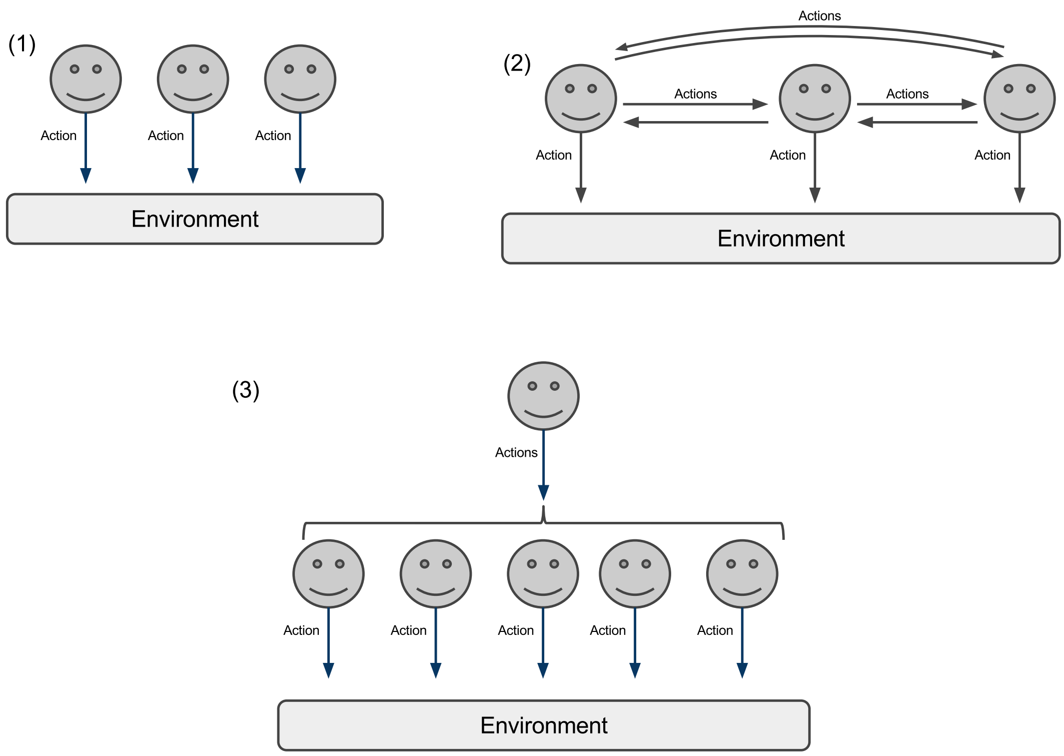


Figure 1.7: Some examples of organization features: (1) Knowledge of other agents (2) An agent communicates with other, (3) Hierarchical

require rules to interact between all the distributed components. Such interactions create networks that share important information, enable performing collective work and making collective decisions. Thus, the system performance has a collective behavior that emerges from the interaction performance between each individual component. The complex systems are frequently adaptations of existing animal and human society working structures. In the context of this thesis we consider the MAS approach as a way to create solutions under the complex systems philosophy.

1.4.1 Emergence general context

In the nature the emergence is a natural way to find solutions to problems (Holland, 1995). For instance the bees have found the best way to store honey using hexagonal shaped cells, as long as we know the bees are not engineers and they have not an advanced brain capable to make such complex decisions. To understand this we can analyze briefly they behavior: each bee takes a drop of honey at a time. Thus they put such drop one next to each other to fill the panel of cells. We realize that the shape of a drop is circular but when several drops are gathered around one all are compressed to form a hexagon. Such shape is considered the most efficient shape to store such nectar (See fig. 1.8).

However for (Holland, 1998) the emergence approach is defined through the employment of algorithms based on rules (similar as game rules) and arrays that stores experience values. These

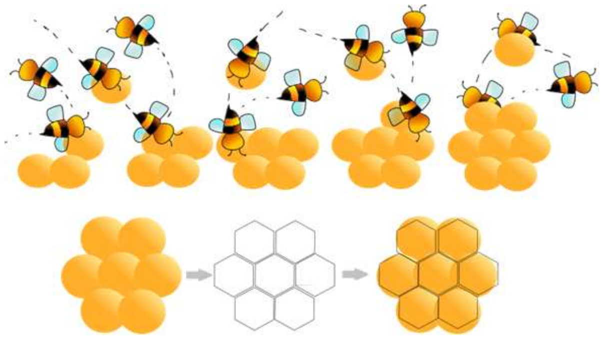


Figure 1.8: Bee's emergent behavior reflected in the hexagon shaped cells.

rules act similarly as the physic rules in the nature and the learning and experience are more like the evolution and adaptation of living organisms. When the algorithm executes and interacts with the user or environment such arrays are updated. Therefore each time the algorithm has a better performance, thus, emergently it evolves and becomes more and more intelligent or adapted.

The emergence from a popular culture point of view given by (Johnson, 2002) is a collective behavior that arises from communities of all levels in all the living beings in nature. However it pushes to enterprises, institutions, communities, neighborhoods, etc to identify such emergence collective behavior to make decisions and profit such situations.

Along the time have existed a lot of kind of bees and only the species that have adopted such behavior are the only ones that have survived to this day. That means that by evolution the most suitable solution has been found and it emerges from a simple behavior. Therefore the emergence concept for us means a simple set of behaviors that combined are helpful to solve a complex problem.

1.4.2 Collective intelligence

The concept of intelligence drives to discuss about the Internet and the use of it as a media to share and find different kinds of knowledge. Using search engines like Google, Bing or Yahoo every one can find a lot of information. Thus we consider that the Internet is the biggest media to find such collective intelligence.

Furthermore we can locate different methods that require a knowledge base updated by people around the world to work. For instance, the meta-analysis in medical research requires a knowledge base of results from trials. Such knowledge base is updated by medical related personnel around the world. Such updates comprise results of trials about treatments studies of different illness and symptoms. Therefore these concentrate of results are commonly employed as source of knowledge. Hence, when a doctor looks for studies related to the employment of a medicament or the treatment of any illness he can look for such related trials at Medline and the literature, thus, extract the relevant results and perform the meta-analysis method to make decisions. In such case the collective intelligence is ordered in a manner that everybody can use them to make decisions, nevertheless, we can find several and different situation using such kind of intelligence.

1.4.3 Self-organization

An organization from a static point of view acts like a structure that describes the interactions and relationships between its individuals (Fox, 1988). It can also be considered as a structure of communication, cooperation and coordination. Nevertheless from a dynamic point of view it represents the elaboration process of such a structure and the same process result (Malone, 1987). Therefore the self organization is the system ability to organize itself through their entities interactions. Commonly the self organization concerns to a spontaneous, dynamically created organization. According to (Di Marzo Serugendo et al., 2006) there are many self organization definitions that corresponds to the different self-organization behaviors :

- Swarm Intelligence. Within the mechanisms related to the swarms behaviors are (Bonabeau et al., 1999):
  - Multiple interactions among the individuals.
  - Retroactive positive feedback that increases the pheromone level when reward is received.
  - Retroactive negative feedback that decreases the pheromone level.
  - Increase of behavior modification that increases the pheromone level when a new route has been detected.
- Decrease of entropy In (Glansdorff and Prigogine, 1971) are shown a set of four prerequisites for systems owning a self-organizing behavior with external pressure.
  - Mutual Causality: Where at least two of the system components have a circular relationship that influences one to one in both senses.
  - Auto catalysis: It rises when at least one of the system components is resulting in its own increase causally influenced by another component.
  - Far-from equilibrium condition: It appears when a large amount of energy is imported from outside of the system. Therefore the system uses such an energy renewing its own structures (similar as autopoiesis) dissipating instead of accumulating, the accruing disorder (entropy) return within the environment.
  - Morphogenetic changes: It occurs when at least one of the system components is open to system external random variations and they change themselves.
- Autopoiesis. A system within this category is an organised unity that is composed by a network of transformation and destruction processes of components production. The produced components commonly:
  - 1. Regenerate and realize continously the network of processes or relations that produced them through their interactions and transformations.

  - 2. Specifies the topological domain of its realization as a network building the machine as a concrete unity in the space which the components exist.

- Artificial Systems. In (Serugendo et al., 2004) two definitions of self-organizing systems have been established:

  - 1. Strong self-organizing systems are considered as systems that modify their organization without any central control: explicit, internal or external.
  - 2. Weak self-organizing systems are considered as systems where reorganization emerges from an internal central planning or control.

  In this context, self-organization relies on ordered structures and component behaviors. Moreover, the self-organization process modifies the structures and behaviors in order to build a new distinct self-remade organization.

- Self-organization vs emergence. Emergence is considered as the situation when a structure, that is not explicitly represented at a lower level, appears or emerges at a higher level. The notion of emergent properties comes from the case of dynamic self-organizing systems, with decentralized control and local interactions. As instance the case of the ants that establish the shortest path between the nest and the source of food. Nevertheless from the perspective of self-organization it can be seen without emergence and vice-versa.

Self-organization has been studied within different domains (Bonabeau et al., 1999):

- Biology. Using the insects and animals behaviors like a path to create self-organized solutions.

- Thermodynamic. Employment of the physic properties to create chemical components.

- Cybernetic. For instance the Neural Networks, and Genetic Programing.

Linked to the MAS area we can find several works related to the self-organization. Such works commonly seek to solve a problem in a distributed manner and their main build task is to ask questions about the organization. For instance considering an open MAS it needs to have the means to adapt its organization to reach its objectives in the best manner.

Indeed, if an agent had a specific role within an organization at the moment when it leaves such an organization, the system must find someone to replace it or assign to other agents the tasks that occupied by the leaving agent. Thus the system adapts dynamically to the agents organizational structure to reorganize work.

There are some approaches like (Drogoul and Collinot, 1998a) proposing to use a group organized as a teamwork or proposing a form of collective intelligence (Trianni et al., 2008). Swarm of Robots, group of agents, etc. are employed to perform collective tasks. Individually they usually implements behaviors inspired in the natural behaviors of ants, bees, dragon flies and birds.

Therefore the collective behavior emerges from interactions of such individual behaviors. Commonly these collective behaviors are capable to find shortest paths, create group formations, visual communication and so on.

There are also works inspired in human nature like (Minsky, 1986) that proposes a human learning organization based on a society of agents. Where such mind agents acts together to learn, remember, think, perform complex tasks and achieve collective objectives.

Commonly the self-organizations focus on three main points (Marcenac et al., 1998):

- The detection of the best conditions for the appearance of the self-organization phenomenon. It is related to the finding task of the different parameters that influence it appearance. This task is distributed over the different components of self-organization. This is therefore at the level of the organization, to find agents who are in a state called unstable, in other words, agents whose aspirations are defined by the owned roles in the organization are not yet achieved. Thus the agents that share characteristics are grouped together and they share messages with other ungrouped agents, under the organization reconstruction objective, in order to invite them to participate in the organization emerging phenomenon.

- The emergence of the phenomenon: this mechanism is responsible for aggregating the agents and to ensure to keep the adaptation of the organizational structure. The messaging sharing between agents leads to the conformation of society organization, the establishment of relationships between agents and groups creation.

- Stopping the development of the phenomenon requires to identify agents that are stable and is therefore necessary to make observations. The observations are commonly recognized between three types: (1) External MAS level, when we consider the MAS as a black box and we can only observe inputs and outcomes. (2) MAS Internal level, when we observe directly at the MAS agent's interactions level. (3) Agent Internal level, when we consider the agents interactions and their internal architecture.

1.4.3.1 Mechanisms

A brief list including some of the main self-organizations techniques is described in the next paragraphs:

- Self-organization skills with reflexes. Commonly in this self-organization mechanism, the agents are reactive and them adapt the performance of actions based on their environment perception. In this way each organization agent specializes increasingly. Such a specialization phenomenon is discussed in (Marcelpoil et al., 1994) that illustrates the specialization of agents acting like "cells" based on available resources. Another interesting approach is introduced in (Ray, 2011) considering a self-organization algorithm based on the the fruit flies nervous system cells dynamic self-organization.

- Self-organization by monitoring interactions. This type of self-organization mechanism is commonly applied using agents reactives and it works adjusting the mechanisms of interaction patterns. For instance, the PACO cases (patterns of coordination (Demazeau, 1993)) where the interactions include the perception of the environment and the interactions with other agents.

- The W-learning and Q-learning. The W-learning mechanism created by (Humphrys, 1995) allows agents (or autonomous robots) to learn which are the actions to be triggered considering what are they perceiving from their environments. Therefore the agents always chose the action to perform considering which could be the most relevant result. A state feedback allows the agent to adjust the importance associated with the action and quality. Such proposition is introduced by the Q-learning algorithm (Watkins, 1989; Humphrys, 1995)).

- Self-organization with preferential links. It consists in multi-agent systems where the agents know each-other. It enables to choose the specific agent with whom to interact according to their offer of services. The historic performance of previous interactions can help to make these choices. For instance, the preferences on such relations could be based on a varied criteria as response time to a query or data accuracy. In (Francis and Heylighen, 1996) a mechanism is employed to improve the efficiency of the Internet employing learning functions. Such functions consider the links most commonly used by the Internet users by creating an associative arc using a weight. Therefore the hyper links with heavier arc weights are the most often offered to users.

- Relaxation. This mechanism belongs to the self-organizations mechanisms based on collective learning acts. The basic mechanism, paradigm of relaxation, was proposed by Les Gasser (Beck, 1994). Such mechanism proposes that the employed knowledge passes through the interactions that will be stored and reused (Camps and Gleizes, 1995). Commonly an agent that is interested or not in a message could store it or not depending on the relevance of the message. Thereafter, the agent will choose to retransmit the message to agents that can be interested in the contents of this message or satisfy the message request. During this operation the agent lets the id of the agent that originated the message. However if it finds the information he received unnecessary or false, it can destroy it.

- Self-organization and reflexivity. This mechanism organization relies on the fact that an agent could need to organize a group of other individuals in order to accomplish a task. In this case, it plays a dual role as it should: (1) Choose an organization to impose lower hierarchy agents. (2) Make an organization emerge through messaging contact and negotiation with agents that belong to the same organizational level in order to share their task or work to perform. The agents are organized to get organized as stated in (MARCIA-(Group), 1996), therefore the cooperation between agents are replaced by cooperation between collective organizations.

- Self-organization by changing roles. In this mechanism the roles of agents are the key feature. Commonly there is a hierarchy that is built from the messages exchanged by the agents. This is similar to contract network proposed by (FIPA, 2002) where an agent acts as the contractor and other as the contracted over different contracts. It is considered as a protocol that allows the agents to organize themselves into small groups through the relations contractor/-contracted.

- Self-organization by instantiation of organizational structures. Such a mechanism is directed by the choice of a priori known organizations. In (Dignum et al., 2004) is discussed the development of reorganization issues within multi-agent systems, specifically where agents determine which of several models of known organizations seem best suited to the situation they face and then apply it. The work of (So and Durfee, 1993) also point towards this direction. Indeed, the system examines the properties of several organizational structures and determines that it should be applied according to the situation encountered criteria.

- Self-organization with introspection. The introspection mechanisms are held within an action considered as centralized where an entity can be seen itself. It works recording traces of interaction actions while the agents are solving a problem and drawing conclusions to improve the organization. This implies the existence of a level base and a meta-level. Their operations are (1) the reification, which enables going from level base to meta-level, and (2) the denotation that is its dual (inverse) operation. However an agent can use two types of introspection: (1) The physical introspection allows to verify the functional integrity of a multi-agent system in terms of distribution of the workload, reducing communications costs etc. (2) The cognitive introspection quantifies the use of certain services that the agent offers the workload task, etc. This information, combined, allow the agent to know or not whether to reorganize their skills in order to improve their performance.

- Self-organization by sharing knowledge. Such self-organization is based on the reconstruction of the agents knowledge: the agents are decomposed when their workload becomes too big. It conducts towards a tasks parallelization and thus it improves the system performance even more to cooperate with other agents to free the resources used equipment. In (Ishida et al., 1992) the application of self-organization in a system composed of several problem solvers is addressed. These solvers can be considered as production systems and some rules are dependent on and interfere. It may therefore be needed for the agents to synchronize in order to maintain the consistency of their data.

- Self-organization based on cooperation. According to (Di Marzo Serugendo et al., 2006) the self-organization mechanisms based on cooperation behavior, permit to treat with applications using continuous or discontinuous global behavior. This kind of self-organization commonly has a bottom-up design that simplifies the development and the resulting systems are robust and adaptive. The AMAS theory states that the system only adapts its behavior to be cooperative with its environment and to satisfy it. Therefore the main difficulty is located in the long list composed by all the non cooperative situations that an agent could find. However, in theory it is viable because the total amount of non cooperative situations related to the agent skills is enumerable.

- Some hybrid mechanisms work up with several of these mechanisms making possible to mix different self-organization mechanisms in order to achieve an objective. For instance in (Frederic, 1996) mix mechanisms for sharing knowledge and introspection where the decomposition phenomenon appears when the mechanism of introspection assumes that an agent needs help (in reason of an overload of work, time constraints, etc) considering the result of the application.

1.5 Discussion: Emergent collective behavior on MAS

In order to perform an emergent collective behavior within a multi-agent systems we must consider the outcome and how it could be linked to each agent individual performance. In our approach we have identified several issues related to the problem that we are trying to solve that has make us think in a MAS with emerging collective behavior. Using a group of agents, we have recreated an artificial group of meta-analysis specialists to build possible solutions making individual decisions. Also we have proposed to employ a meta-model perspective to take such decisions, in other words, the meta-model group solution is built depending on the meta-model owned by an individual agent.

Chapter 2

MAS Related Software Engineering

There are several methodologies managing the analysis and development of MAS. These methodologies are considered part of the Agent Oriented Software Engineering AOSE. Furthermore in section 2.5 we state a survey about the relation between existing MAS methodologies and our contribution.

We have found few related approaches with some similar objectives; such approaches consider some issues that are similar to ours. We can cite them as a starting point from the context of MAS Decomposition, model-driven engineering (MDE) and reuse of existing methodologies:

2.1 Background

The multi-agent concept appears to be an innovating and efficient paradigm for modeling of complex artificial systems. The multi-agent approach descends from diverse biologic and social theories or from the decision. The multi-agent entities may consist of numerous types of capacities, from the most reactive to the most cognitive ones. These entities are grouped within certain models making functional the agent architectures, the interaction and perception capacities, the treatment of environment. From the social point of view the multi-agent devices are integrated by diverse interaction models and organization structures linked with the agents' nature.
Numerous multi-agent methods which exist nowadays propose the use of the problem engineering within specific domains. The software development cycle traditionally consists of at least two phases: analysis and design. If we plan to base our work on the multi-agent engineering, we will distinguish two corresponding development phases, which can be called Multi-Agent Oriented Analysis and Design.

2.2 The use of Meta-models or Components within the methodologies

Among all the MAS methodologies we have found some approaches that propose to employ a component or meta-model based process. The next table shows these approaches an their relevance:

Meta-models in Methods			
Method	Meta-models	Relevance	Comments
ADELFE	Yes	It is explained considering the features that a cooperative agent possesses. The life cycle consists in having perceptions, making decisions and then acting. It considers also the Non-cooperative situations	It is relevant because it proposes directly a reusable meta-model that can be characterized.
Extended Gaia	Yes	It focuses on social aspects and organizational structure. It proposes to accurately specify the relationships between different entities within a specific organization context. It employs the classic Gaia building blocks: agents, roles, activities, services and protocols	This is an extended version of Gaia that introduces a meta-model based on the organizational structure.
INGENIAS - Message	Yes	It employs five meta-models: Agent, Interaction, Tasks, Environment and Organization. Each meta-model describes the corresponding types of diagrams where an instance of a meta-entity could be situated in different diagrams. Each diagram can describe a system from the point of view.	It also is near of our approach employing different kind of meta-models nevertheless it describe the system from the meta-model kind point of view.

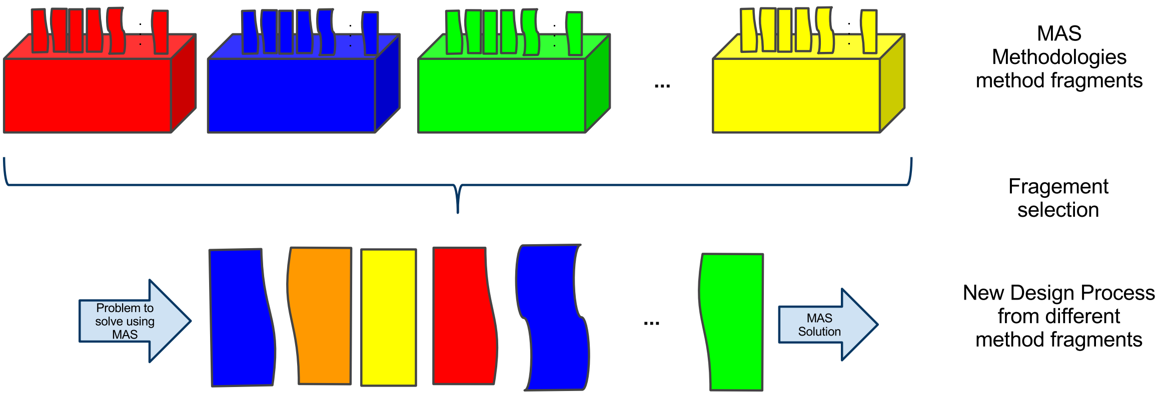


Figure 2.1: MAS Method Fragments approach concept, the figure show the main idea of method fragments where different method parts are taken from different methodologies to build a new design process.

PASSI	Yes	It is splitted into three domains: Problem, Agency and Solution. It proposes to identify resources, requirements and scenario from the problem and thus connect them with Agent Role and Tasks. Specifically it proposes go from the requirements to the code (implementation)	PASSI is the approach that is nearest to this thesis contributions because it proposes to identify the requirements domain thus connect it with the agents and then with the implementation.

Table 2.1: Comparative summary of methodologies Meta-Models

2.3 MAS Decomposition

There are some approaches that proposes to divide a MAS into parts. In (Mariachiara et al., 2010) is proposed to divide the MAS methodologies process and then compose a development process employing such parts. Also in (Seidita et al., 2006) is proposed to use a scheme based on fragments that are obtained from different software engineering design processes from existing MAS methodologies. This enables combining stages of different design process in an appropriate way to conceive MAS-based solutions that would otherwise not be obtained. The resulting deliverable is based on UML, diagrams and MAS global meta-model [1].

There are also approaches that focus over some key parts of a MAS as a central aspect to design a MAS. Some of such approaches are Agent Centered MAS (ACMAS) and Organization Centered MAS (OCMAS) (Ferber et al., 2003) (See fig. 2.1). Where each one centralizes the development

[1] A meta-model is a model description or model of models (Z.Guessoum and Jarraya, 2005)

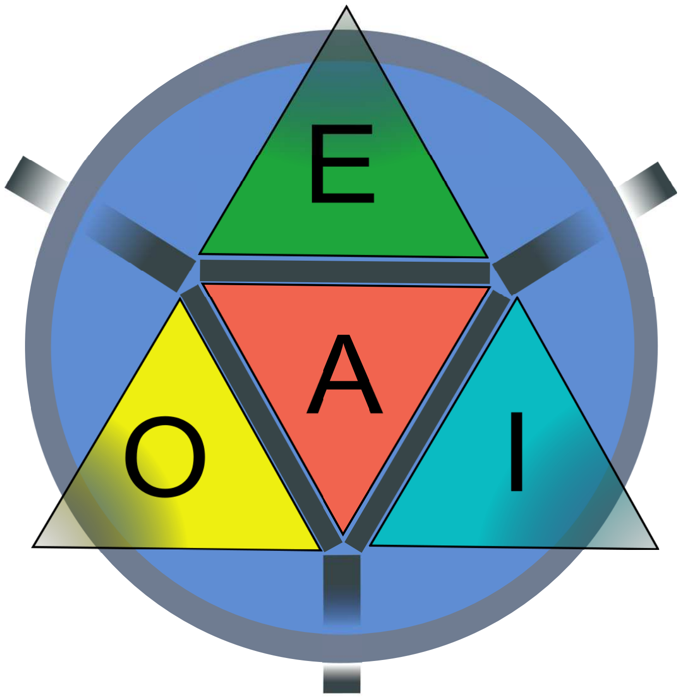


Figure 2.2: AEIO Vowels MAS approach concept, we can see the four types of components that builds Multi-agent systems: Agent, Environment, Interaction and Organization. All linked by gray wrappers and unified by a blue globe that represents the Unification implicit component.

using as key design part the agent or the organizational aspects.

There are also fragments approaches with the perspective of building every MAS from different parts.

The AEIO vowels approach (Demazeau, 1995) is based in the decomposition of every MAS into four main type parts where each parts begins with a vowel letter: Agent, Environment, Interaction and Organization (See fig. 2.2).

The AEIO decomposition objective is to create components with characteristics that enable reutilization and easy refactoring of them into the MAS creation process.

Our approach instead aspires to use meta-models as fractions of a solution, such as proposed in (Rougemaille et al., 2008), but using a format based on vowels (Demazeau, 1995). Furthermore (Rougemaille et al., 2008) proposes to use meta-models based on several existing methodologies (Gómez et al., 2007) but this makes it dependent on a format it does not take into account a combined use of meta-models derived from different methodologies to build one solution. As we state, in (Seidita et al., 2006), is considered a combination of software engineering phases, however, our approach proposes the use of meta-models from different methodologies to be used in combination into a solution.

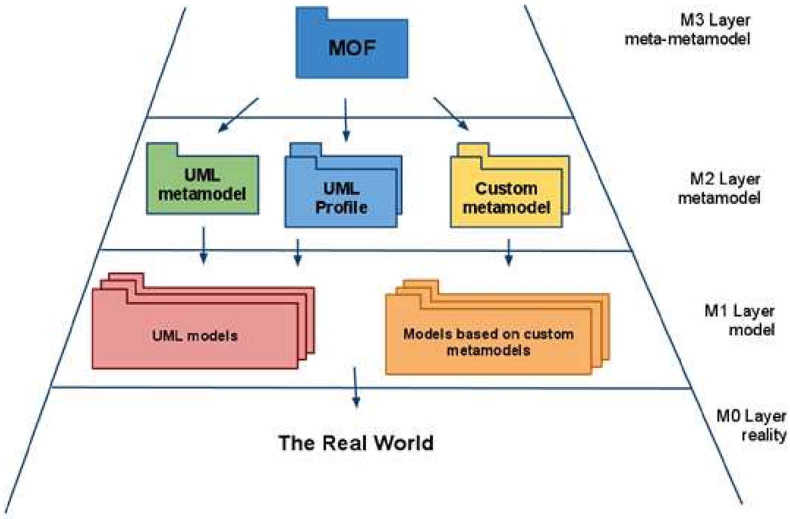


Figure 2.3: Four Layer Model-Driven Architecture overview

2.4 Model-Driven Engineering

The model-driven engineering approach has a high significance in the context of software engineering. It has evolved from the object oriented paradigm, so for the MDE everything is considered has a model instead as an object. MDE usually manages two types of relations: representations and conformances. The representations are model of the real world has software artifacts for example, The conformances are models created conform a meta-model. Thereby the MDE is considered the disciplined and rationalized production of models. For the present thesis the conformance relation has a relevant importance because we propose the meta-models employment. The models under the MDE philosophy usually are constructed under the next characteristics structure (Selic, 2003): Abstraction, understandability, accuracy, predictiveness and inexpensiveness. The model-driven engineering theory specifies the rules and baselines to work with different kinds of models. In (Gasevic et al., 2009a) is described the use of a mega-model which defines physical, abstract and digital systems. For our purposes we focus on the digital systems, specifically on software systems.

Finally we must mention that the present work uses only a characterization that comes from meta-models of the M2 Layer. Such meta-models are adapted from the Agent Oriented Software Engineering, this concept will be described further.

2.5 MAS Methodologies Context

Approaches as the following: Gaia (Wooldridge et al., 2000), DIAMOND (Jamont and Occello, 2007), PASSI (Cossentino, 2005) and Tropos(P. Giorgini, 2004; Castro et al., 2005) cover different aspects to solve a variety of problems centered in delimited domains. Gaia is a general methodology that sets the goal of understanding the system and its structure, this, without any reference to the implementation. In addition, it is oriented only to closed domains where the features of MAS are static. However, with the use of extensions (Garcia-Ojeda et al., 2004; Wei Huang, 2007) Gaia can be applied in dynamic domains. In contrast ADELFE (Bernon et al., 2002b) focuses on problems that can only be resolved through AMAS (Adaptive Multi-Agent Systems). DIAMOND(Jamont and Occello, 2007) is a method that enables to generate embedded MAS. Also ADELFE bases its guideline on the RUP (I. Jacobson, 1999) adding a design work flow, agent model and NCS model (Bernon et al., 2002a) to guide the developer in the creation of a MAS. So DIAMOND uses various linked stages to develop embedded MAS considering the software and hardware at the same time. In addition it offers a way to create reusable components by identifying and delimiting the domain of the problem within one of these stages. Also DIAMOND defines a life cycle for hybrid software and hardware MAS. The process PASSI (Cossentino, 2005) guides the development of multi-agent software step by step from the requirements to the code. All these methods integrate design models based on UML (OMG, 2003) notation and different philosophies regarding object-oriented software engineering for MAS (J. Odell, 2000; O. Gutknecht, 1999; Shoham, 1993; Thomas, 1993; Caire et al., 2001b).

The methodologies frequently base the software engineering application on different models and diagrams derived from UML. Their objective is to develop different and specific MAS. The use of models derived from UML (OMG, 2003) is located in the M1 layer of the four-layer Model-Driven Engineering (MDE) (Kleppe et al., 2003). It means that these models belong to a specific meta-model (Zhang and David, 2005)(Ferber and Gutknecht, 1998a). Each model is designed to solve problems for a specific aspect, such as agent creation or communication.

Therefore there are several methods that we have considered in order to analyze the current state of development in relation with the pre-initial phases in the MAS life cycle of software development. Considering such issues as the key features to analyze we have considered ADELFE (Bernon et al., 2005) as an adaptive MAS domain-oriented methodology that focalizes its initial steps giving standard software engineering options based on RUP and UML it also recently added a initial phase where the designer can check if their problem is able to be solved using the AMAS approach.Aalaadin described in (Ferber and Gutknecht, 1997) is an Organization Centered MAS (OCMAS) methodology that assumes that the designers problem could be solved using such development view. Cassiopeia (Drogoul and Collinot, 1998b) at the initial phase identifies the roles to be performed within the MAS, it states that such approach is employed for team work, therefore, the designer must know that their problem domain belongs to such kind of solution. DESIRE (Brazier et al., 1997) propose to decompose the problem in agent tasks and assumes that the designer know that it works well for their problem. DIAMOND (Jamont, 2005) belongs to the embedded MAS domain but the designer must choose if their problem needs such method. Gaia (Wooldridge et al., 2000) is in a closed domain with statistic features therefore the designer must situate their problem within this domain before choosing Gaia. MAMOSACO (Adam, 2005) focuses on management complex systems, therefore before choosing this method the designer must identify their problem as resoluble within it. MaSE (Deloach, 2001) centers it attention in the goals, then the designer must analyze their problem to locate it on this method. MASSIVE (Jürgen, 2001) proposes to develop from different views enabling to see the problem solution from different perspectives, however it is aspect oriented and the designer must choose the method considering that their problem is compatible with it. INGENIAS-MESSAGE (Pavón and Gómez-Sanz, 2002) employs meta-models and views to solve the problem but it does not propose how to match the designer problem with the method domain. PASSI (Cossentino, 2005) proposes to identify the requirements domain in order to the design a MAS using such a method, however, it does not propose how to identify such domain. Prometheus (Padgham and Winikoff, 2004)proposes the reusability prototyping agents, moreover, it focuses on the industry, scheduling and debugging agents. Therefore it does not define how a designer can situate their problem within its domain. Tropos (Giorgini, 2004) proposes to analyze the early requirements in order to identify a domain, (such a situation that we have also considerd for our approach), nevertheless it focalizes only in the Tropos method and does not propose reusability. The Vowells (Demazeau, 2001) methodology proposes MAS decomposition and reusability, nevertheless it does not solve how to identify the best components to create a MAS based solution. The table 2.2 lists some of the main MAS methodologies and summarizes their relationship with a pre-initial phase.

Life Cycle				
Method	Type	Goal	Pre Initial Phase	Comments
ADELFE	Cyclic	Adaptive MAS	No it only helps giving a UML and RUP based initial phase, it recently added a prealable phase where the designer can verify if their problem can be solved using AMAS approach	Uses a work flow based on RUP, It proposes to identify interaction as key feature for Adaptive MAS development
Aalaadin	Waterfall	Organization and Roles Centered MAS	No, it assumes that the designer knows that their problem can be solved using and OC-MAS	Centralized in Agent, Group and Role development it proposes a general MAS meta-model

Cassiopeia	Iterative	Dynamic Organizations and Role like robots soccer	No at all, but it proposes as a key initial feature identify the agent roles in order to apply the method,	Centralized in Agent with three kind of roles: domain-dependent, relational and organizational. It also proposes two kind of organization: static and dynamic.
DESIRE	Waterfall	Agent Task Composition Centered	No, only task decomposition the designer must know that their problem is ad-hoc with the method	Proposes build a MAS employing a task (de)composition, information exchange, sequencing of (sub)tasks, subtask delegation, and knowledge structures.
DIAMOND	Spiral	Embedded MAS	No, but it consider that the designer must develop an software/hardware mixed MAS	It proposes to consider both developments software and hardware in the same development cycle in order to build an embedded MAS. However to build a MAS it considers individual and society levels.
Gaia	Iterative	General MAS, role centered	No, because its a general MAS it assumes that every problem could be developed using Gaia. However it is oriented to a closed domain with static characteristics.	It has two main phases: analysis and design. We have a set of models as result of applying this methodology that enables a designer to begin the implementation.

MAMOSACO	Iterative	Complex management systems	No, but it focus on complex management systems	This method considers the complex managements systems and comprises the phases of analysis, modelization, specification and conception of such systems.
MaSE	Iterative	Goals centered	No, however it guides the designer through the software life cycle at the beginning its supposed to be compatible with the designer problem to solve. Its development process is based on RUP. Also it has automatic code generation using AGENTTOOL.	It is independent of MAS and agent architecture, programming language or programming language
MASSIVE	Iterative	Aspect oriented, View system	No, it has only different development views in order to develop a MAS using different views. Also it proposed to link model features with implementation features	It is composed of the following views: Task, Environment, Role, Interaction, Society, Architectural, System

INGENIAS - MESSAGE	Iterative	Model-Driven Oriented with several development views	No, but it employs meta-models to build solutions.	INGENIAS is based on MESSAGE, currently it presents a set of meta-models aspiring to analyze all the system using the language GOPRR (Graph, Object, Property, Role and Relationship). It has the following development views: Organization, Agent, Tasks/Goals, Interactions and Environment.
PASSI	Incremental		No at all, it only states that it need to know the domain requirements description at the beginning	It is a step-by-step requirement-to-code methodology for designing and developing multi-agent societies, it comprises five model phases: System requirements, agent society, agent implementation, code and deployment,

Prometheus	Waterfall	Goals centered, Covers mainly domains of industry, scheduling or debugging agent systems	No, but it enables to reuse and share different MAS models and create prototypical core agents	It is a methodology that guides from the start to the end and enables to the developers to analyze, design and implement MAS prototypes. Basically, it consists of three phases: System specification, Architectural and Detailed design, It generate automatically code under the JACK programming language.
Tropos	Incremental	General MAS	No at all, but it propose to analyze the early requirements to gather early concepts based on i*(Siu-Kwong, 1996) in order to identify a domain	It is based on two main ideas: (1) the notion of agent and all related mentalistic notions: goals and plans for example. These ideas are employed in all phases of software development: from early analysis toward the actual implementation. (2) It also covers the very early phases of requirements analysis, therefore it enables a deeper software environment understanding, and identify the kind of interactions between human agents and software

Vowels	Waterfall	Component centered	No, but we consider it as the best way to decompose a MAS trough the AEIO axes, similarly as other methodologies perform using views	This approach proposes to identify a MAS from four axes: Agent, Environment, Interaction and Organization. Considering each one of the axes independently in order to reuse each different kind of component but it also proposes to unify all the components to make them able to work together using wrappers and compatibility rules.

Table 2.2: Comparative summary of methods life cycle coverage

2.6 Discussion: The difficult methodology choice and the impossible unique approach

The models defined in each platform are different, for instance, an agent model defined in PASSI (Burrafato and Cossentino, 2002) could not be used in ADELFE (Bernon et al., 2002a). But how could we solve a problem that requires a PASSI (Cossentino, 2005) agent model type, which is running embedded according to DIAMOND (Jamont and Occello, 2007) specifications? In the next lines we explain how our approach contributes to the solution of this kind of problems using a meta-model definition that belongs to the M2 layer of the four-layer MDE (Kleppe et al., 2003).

It has been observed that all the afore-mentioned methodologies provide solutions focused on a specific kind of problems and domains. On the other hand, it is difficult to decide in advance which methodology would be appropriate to solve a specific problem. In other words, the domain covered by each method is limited, for example, ADELFE is geared only to the domain of adaptive multi-agent systems and Gaia is directed to a closed domain with multi-agent static characteristics.

The truth is that we can't solve all the possible problems using a single method, because each method covers a limited domain, in which only problems suitable for this domain can be solved. Therefore, a problem that can be solved with one method efficiently probably cannot be solved with another.For this reason our approach proposes to establish a preliminary stage, in which the problem is analyzed to determine its domain. This verdict guides the developer stating which would be the best way to solve this problem under the predefined selection criteria. Besides our approach defines the use of meta-models based on the AEIO vowel Decomposition (Demazeau, 2001; Occello et al., 2004; Ricordel and Demazeau, 2002) for MAS approach. The main objective of these meta-models is the reutilization and adaptation of the different models that already exist, providing an option to make MAS. This aspect is similar to building blocks or design patterns(Holland, 1995), but our approach will neither change the existing models in other methodologies, nor propose a new model standard.

It is difficult to propose a unified global approach to multi-agent systems engineering, because the problems and domains addressed by this approach are highly heterogeneous. The nature of the problems for which multi-agent systems are effective, such as simulation, problem solving, integration of applications and system or management systems, provokes that the use of simple models impossible.

The domains in which these problems are solved impose the use of specialized embodied approaches. How can we unify the design of remote applications such as Internet Web services or production systems? The diversity of methods makes the choice of approach difficult for the non-specialist designer. A comprehensive preliminary analysis phase lacks, which is capable to unify the model specification choices, depending on the domain and problem of the target software.

We have a lot of methodologies that provides a wide range of options to develop a MAS-based solutions also each methodology provides process, components and tools to develop the desired MAS. Nevertheless, if the designer is not experienced with these methodologies and he does not know at all the components provided, it is difficult for him to choose the most suitable methodology. Also an inherent problem of the MAS methodologies is the different-methodology-components usability because we can't take components from different methodologies to create MAS-based solutions easily; and finally the reusability of already created solutions. Consequently, we propose to abstract the MAS components using the AEIO vowel approach (Demazeau, 2001) from the MAS methodologies to create MAS Meta-models for each component. The aspiration of this is to have different kind of Agents, Environments, Interactions, Organization and rules of compatibility and unifications between all these kinds of components. Going further we propose to characterize them and evaluate each characteristic of them dynamically considering a link between each characteristic and a domain of solution. That means that each AEIO vowel meta-model will have a set of characteristics with domain efficacy values that will be used for meta-analysis.

This proposal should not be reduced to only use the MDA approach (Model-Driven Architecture) (Kleppe et al., 2003) modified for the MAS engineering. We are not focused on the use of meta-models to transform them into models of implementation in the sense adopted by MetaDIMA (Z.Guessoum and Jarraya, 2005), ADELFE (Bernon et al., 2005) and PASSI. This technique can be used for detailed analysis, but as already mentioned, we want to work on an preliminary conceptual analysis.

Our goal is not to unify the different meta-models into one as proposed in the work proposed by (Bernon et al., 2004). We cannot adopt a single meta-model as in (Knublauch and Rose, 2002). It is impossible taking into account all in only one meta-model because of the difficulties at the implementation and at the deployment in a different way in terms of domains. Instead, to describe our meta-model (analysis model) we could use a Meta-Meta-model[2] as proposed by the method MESSAGE / AUML (Caire et al., 2001a) (it uses the MOF UML).

[2]A meta-meta-model is a model that enables to model or describe meta-models, moreover, a meta-model is a model of model.

Chapter 3

Meta-analysis and Decision Making

In this chapter we expose two main fundamentals of this thesis: Meta analysis and Decision Making. Each one has a different contribution but both share the fact that helps in the complicated task of predict a result. The meta-analysis provides a successful methodology that uses statistical data in order to show a tendency and thus make decisions. The decision making is addressed within a MAS state of art, however, its contribution lies in providing the algorithms to automate the decision making. The algorithms that are presented belongs to the Bayesian Cognition (See section 3.2.1) and Decision Making context (See section 3.2.2).

Both contributes directly in the backbone of the prototype: The agent individual decision making. It contribution within the statistical data gathering and induction methods is also related with the future work that is addressed in the section 9.2.

3.1 Meta-analysis

Commonly, the meta-analysis is performed for one or more researches and they must look for related studies in the on-line electronic databases or the paper literature. The researchers must have enough experience to select related trials, it means, to select trials with studies using similar trial protocols and compatible outcome: for example selecting the studies that evaluate the efficacy of a new substance or new medicament against cancer versus a placebo or an old medicament. Once they have found an adequate number of studies the next step is to group all the studies trial results using the meta-analysis common methods - Odds-Ratio and Mantel-Haenszel - using a model - Fixed Effect Models or Random Effects Models - depending on the statistical heterogeneity (Leandro, 2005). Consequently the gathered data is employed to obtain several statistical information, for example, the 95% confidence interval that enables seeing the general studies results tendency and further make decisions using such statistical information (Leandro, 2005). By the way, the meta-analysis needs an enough amount of studies results to reach an effective statistical information and thus a reliable tendency. In addition, each trial is composed by their study results, which ones are created using a trial protocol. Usually a study shows comparative results between medicaments, substances

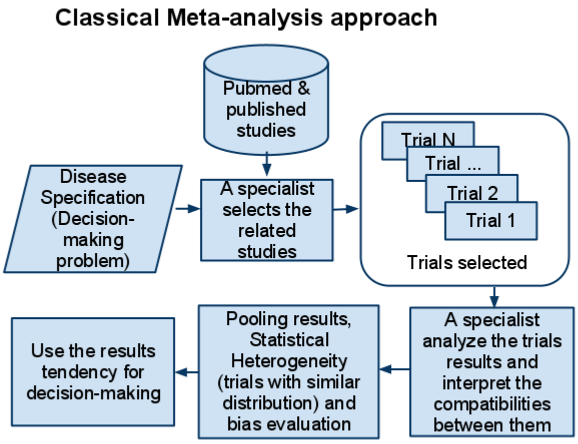


Figure 3.1: Classic meta-analysis process overview

and placebos that show evidence about their efficacy using them against some illness (See also fig. 3.1).

3.1.1 Meta-analysis Introduction

The meta-analysis is defined by the National Council (USA) Library of Medicine as

> "a quantitative method for combining results of independent studies (usually drawn from published literature) and synthesizing summaries and conclusions that can be used to evaluate therapeutic effectiveness, plan new studies, etc. Its application is performed mainly in research and medicine."

In this proposal, we understand the concept of meta-analysis in the sense that this is employed in medicine to evaluate the efficacy of a treatment or medication (Sutton et al., 2000).

This thesis attempts to equate "clinical studies" to "studies of problem areas." Reasoning about meta-knowledge from analysis to find appropriate solutions in terms of meta-models for MAS. Applying meta-analysis in this proposal is at all times an original approach in an engineering analysis phase multi-agent.

Therefore this approach proposes using meta-analysis to identify components (meta-models) that proved to have worked well with previous multi-agent systems solutions for some kinds of problems, and thus design or develop new multi-agent systems solutions for similar problems. We consider that this is an interesting alternative as this approach has been used in the field of medicine with success.

However to adapt a meta-analysis process into a multi-agent systems we need to reuse and split into components that can be evaluated within the process proposed by the meta-analysis. Such approach that enable to abstract from MAS methodologies is introduced in section 2.3.

Moreover our approach is based on an individual making-decision that needs such a meta-analysis like a process to analyze MAS meta-models. For this reason in this section we discuss the meta-analysis process in order to understand how the meta-analysis work within a clinical setting ant thus equate "clinical studies" to "studies of problem areas."

3.1.2 First steps

In the literature (Leandro, 2005)(LaPorte, 2011)(Armitage et al., 2002) the first step defines which trials results could be candidates to be considered by meta-analysis. Therefore performing a meta-analysis needs:

- A primary outcome that will be determined and should be considered in all studies selected for analysis. (Eg. efficiency of some MAS components into a solution, a MAS component estimated compatibility with others, etc). Then one or more secondary outcomes can be useful, especially under specific questions that are not necessarily considered in all eligible studies.
- A key issue is choosing the characteristics of the tests (trials) that you want to select. Similarly it is necessary to consider the largest possible number of studies in a second phase to choose studies that are based on different discriminating factors. For example: number of issues considered, output measurement, criteria for randomization, and so on.
- It is important to define where and how to conduct the search. In medicine the meta-analysis focuses on a couple of on-line search engines, usually Medline and Embase (NIH, 2010).

3.1.3 Gray Literature

It is called gray literature to those published in electronic or paper, produced by the government, universities, businesses and industries because they goal to publish is not to profit. This literature should be considered mainly because their results are not manipulated or concealed, which sometimes such practices turns out to be a common situation in private medicine publications. This is caused by the trade tendency to show that a new drug is more efficient, then more marketable, so only are published studies that have beneficial results for the clinical laboratory that makes the drug and sponsor such studies. This may cause deviations from a meta-analysis can hardly be corrected. Therefore is recommended to take care in this regard.

Although within the framework of multi-agent systems is somewhat inconsistent we can argue that there is no database of studies of all the methodologies and solutions created from these, thus, it causes a similar problem. That is why the basis of meta-analysis strongly suggest the use of all or most of the literature.

3.1.4 Source of troubles

Another source of trouble for conducting a meta-analysis includes:

- The language barrier that prevents all publications are considered even if they are in several languages. Usually only the English works are considered, however, there are a lot of publications in other languages with relevant results that are ignored.
- Duplication of works. It occurs due to the publication of the same work results in multiple articles.
- Conflicting results. Similar works that have different or contrary outcome.
- Poorly prepared summaries. They are summaries with not enough or clear information about the work and outcome.
- Publications with little clarity. Some works show partially the outcome or in a manner that we can get clear enough the meaning of them.
- Latency of publishing work. It refers at the situation when the time needed to get published an article.
- Works with negative outcome unpublished. It refers to a tendency in commercial funded research to only publish the results that show a positive outcome and otherwise are not published.
- Unpublished work. When there are works with relevant results that are not published. Sometimes the publishing task becomes very subjective and some works are not accepted.

It is recommended that each form of publication includes:

- Generic information. It commonly includes the names of authors, belonging institutions, addresses, e-mails.
- Design of the test. It must describe how the test has been conceived. Commonly it describes which protocol has been employed to perform such tests.
- Treatment of the study group and control group. It must define the treatments employed for each group. For instance, a placebo an a new medicament treatments.
- Number of events, number of cases within the two groups mentioned above. They are the real quantities of cases and events within the study and control group.
- Results of the calculation basis used in the meta-analysis. Such results are defined by a meta-analysis method that commonly is choose depending on the data heterogeneity.
- Quality of the score if possible. It defines the score quality, commonly it depends on the protocol employed, the quantity of cases within the groups and the period of application of the treatment.

Also need to be made:

- Statistical procedures. The statistical procedures enable to calculate the study estimation in relation with the meta-analysis.

- Interpretation of results. This is commonly done employing a graphic chart type high and low that shows the OR outcome and the CI of each study. Within such chart graphic category there are a lot of options to show the results. However commonly is considered that the chart graphic shows the tendency of the meta-analysis and such data is useful to make decisions.

This last one crucial aspect of the meta-analysis.

3.1.5 Case control studies

It consists of estimates of risk and probabilities of the case studies that can be applied with some variations in the two following aspects:

- Risk difference (RD), which calculates the difference in the proportion of events observed in both groups of study subjects where you applied the same test. In epidemiological terms, the RD is known as absolute risk reduction (ARR).

- Risk Ratio (RR) refers to the degree to which the frequency of an event may vary in the presence of factor under study compared with the absence of such factor latter.

- Odds ratio (OR) is a measure that is similar to RR because it refers to the estimation of the latter when the event is not frequent ($\leq$ 10%).

If the difference of risk or odds ratio is calculated from a meta-analysis, and therefore from many studies, it is called RD pooled or OR pooled. The following formula:

$D = \frac{\sum w_i d_i}{\sum w_i}$

is the general form of meta-analysis where D is the outcome. Where W_i is the weight of each study which translates into:

$W_i = \frac{1}{V_i}$

where V_i is the variance or difference of the i study outcome. The general formula of the meta-analysis the overall results expressed in terms of D weighted average sum of n studies.

3.1.6 Statistical Procedures

It is very important to known the statistical procedures employing meta-analysis when making the choice of test is needed and justify why. The procedures used in the calculation is divided into two categories:

1. Fixed effects models. These are based assuming that the studies taken as a group gave an estimation of the same treatment effect to that intended effects can be considered as part of the same distribution. We have to main methods:

- Mantel Haenszel Method. It defines the result of the comparison in each study. Because of the advantages of precision and multi-method of Mantel-Haenszel it is regularly used and it is usually found in commercial programs that perform meta-analysis in clinical trials.
- Peto's Method. Based on a modification of the Mantel-Haenszel. In the first stage is obtained calculating the expected value of events in each of the groups is done according to the standard formula of total marginal product divided by N. Where N is the total number of cases of comparative studies. Using the calculation of Peto, the presence of values of 0 in a cell does not affect the calculation, and therefore, the approach is not mandatory.

2. Random effects models. These analysis models do not require the assumption that each study is derived from the same population of individuals and, therefore, all studies can be considered as part of separate populations, each with its own mean value. It is thus that the variability of the estimate may have two sources: in one study and between studies.

   - Quantifying heterogeneity. Consists of evaluating the data to know their degree of heterogeneity and variation.
   - Quantifying publication bias. It calculates the bias according to the issues described in section 3.1.3.

3.1.7 Discussion: Our approach and Meta-analysis relationship

Considering the work proposed in this thesis, their relationship to the meta-analysis is given in the analysis of specialized entities (analogous to drug treatment, etc.) considering the problem origin (analogous to symptoms). On the other hand the main difficulty of matching the two approaches is that there is a knowledge base as similar PubMed in the field of software engineering. That's why this approach, as shown below, uses a different method of inference that could apply to have a greater knowledge base.

Despite this, the process of meta-analysis has inspired the creation of this work and therefore seeks to establish a base in the future that supports a closer approximation to the meta-analysis done in the field of medicine. We will discuss it deeply in the section 5.4.5

3.2 Decision-making

There are two main branches of decision-making MAS (Luck et al., 2005): decision systems and simulation systems. Commonly the MAS simulation systems are built upon some real world area and perform a simulation recreating a situation. Such situation can be evaluated into a controlled virtual environment that allows analyze the situation abstracting data directly from the simulation. Furthermore a simulation requires to define rules to govern its operation (Siebers and Aickelin, 2008).

In contrast in the decision systems the agents make decisions together and applying decision-making collective mechanisms.

Thus there are two options: (1) Decision support systems (DSS), they act like a support tool helping users to make decisions employing the results information obtained (Re and Pius, 2003). (2) Decision making systems (DMS), they automatically make decisions as occurred in home automation or robotics (Bessière et al., 2008).

Our entire approach is located near to the decision systems because we propose to only support the initial engineer's choices. Therefore it is classified as DSS type but can be upgraded to a DMS type. On the other side within the MAS environment the agents act together to reach a collective decision, such decision is proposed as a possible solution outside the MAS, thus the MAS is considered as an internal DMS.

3.2.1 Bayesian Cognition Context

In the Artificial Intelligence field we could find several approaches (Stuart Russell, 2009), Some of them works using some kind of logic through Prolog, others use neural networks to recognize patterns or genetic algorithms to learn, etc. Thus, each approach has been conceived to work with some kind of data or situations, for instance some of them are good for informed search and exploration as genetic algorithms, others, for making inferences using well known rules and evidence as first order logic inference algorithms.

In order to meet the decision-making needs of our approach within a dynamic environment characterized by uncertainty we have analyzed among a set of options of AI. We cannot use logic based approaches because we have to work with a characterized data coming from uncertainty sources, therefore the values are often real numbers representing probabilities. Moreover, from a probabilistic point of view the logic sentences and inference is a closed environment extracted from the reality, considered as fully certain. The neural networks could be employed, nevertheless, when we modify or add a new domain, we have to reset the weight values each time. Doing so, we will lose a lot of time retraining the neural network . Genetic algorithms are an option that could be implemented because it performs well the search of solutions that we are looking for. It uses a gen based approach to look for the best suitable combinations; however, the statistic and probabilistic nature of a meta-analysis make us to select an approach related to such fields instead the genetic one.

Furthermore we have selected the Bayesian Cognition approach because it allows to treat with the uncertainty throughout the decision-making using statistical data. The Bayesian cognition algorithms enables to being modified dynamically, we can add or remove variables and the result varies only being more or less exact. Then, the more information you have is more accurate the inference. In the other hand, we can modify the knowledge base independently of the algorithm. So, we could employ different inference methods without changing the rest; for instance, we could employ the Laplace's succession rule as inference method and after use odds ratio with Mantel-Hanzel method from meta-analysis using the same statistical data. Nevertheless the inference methods efficacy varies, for example, depending on the quantity of data. Finally that's why we state that the Bayesian Cognition approach matches well with our approach.

3.2.2 Bayesian Cognition in Decision-making

The Bayesian cognition approach is commonly employed to implement solutions for decision-making problems within the computer vision and robotic fields. Therefore it has proved to work well into uncertainty environments and to use statistic data gathered through sensors from such environments to make-decisions. For instance, a self-driven car using cameras as sensors, and image analysis that helps to avoid obstacles, take care of imprudent pedestrians and animals crossing the road. Furthermore the gathered data act as experience data that is useful to decision-making.

3.2.3 Bayesian representation and probabilistic reasoning

As explained in (Bessière et al., 2008) commonly a Bayesian program is defined using the next structure:

1. Description. It is a probabilistic model about some phenomenon that is obtained from the next two branches:
   - Specification. Such specification expresses the modeled phenomenon knowledge in the following probabilistic terms:
     a Variables. All the important and known variables related to the phenomenon.
     b Decomposition. It is the joint distribution of the variables. Usually is done using a decomposition that keeps the joint distribution as a product composition of simplified distributions.
     c Forms. To compute the joint distribution we must specify all the distributions appearing in the decomposition with all the possible values for each variable.
   - Identification. It is the learning phase of the probabilistic experience acquisition where the initial data is refined and becomes more accurate at each step.
2. Questions. The questions are defined by branching a variables' set in three subsets: the searched variables (on the left side of the conditioning bar), the known variables (on the right side of the conditioning bar) and the free variables. Such questions must be answered using the decomposition and forms definitions.

3.2.4 Statistical Models and probabilistic reasoning

A statistical model determines, within a Bayesian Cognition Algorithm, which is the classification of the statistical values of the knowledge base (experience) inside a bell curve. At the bell in the top is where we find values with greater uncertainty in the slopes the values with low or high certainty.

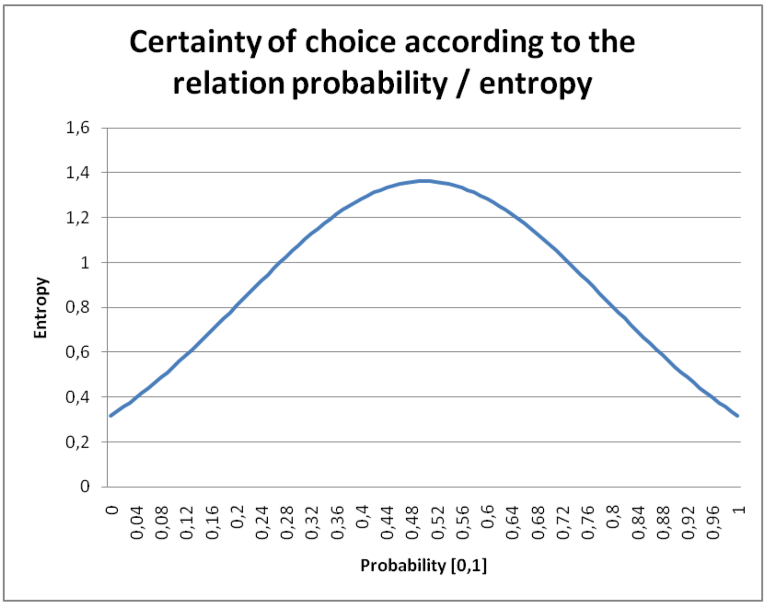


Figure 3.2: Probabilities from 0 to 1 adjusted to a Gauss bell

As we can see in the figure 3.2 a statistical model can be obtained through a Gauss bell adjustment of the values. In such a case the interval of values $[0, 1]$ shows that the values around 0.5 represents the highest entropy or uncertainty. The values nearest that descend to the left from $0, 5$ to 0 owns the lowest likelihood and the values from $0, 5$ to 1 the highest likelihood.

3.2.5 Discussion: Probabilistic reasoning and Meta-analysis in MAS

The Bayesian cognition approach is composed by a Bayesian algorithm that must be implemented according to the related problem. Such implementation is described in section 6.5. Furthermore, we propose the use of such approach within a MAS. Such integration could work into the agent architecture, as we describe in section 6.4.1, as part of the agent cognition skills. Commonly a Bayesian cognition algorithm is employed inside robots that analyze the physical environment and thus make decisions. Therefore, instead a robot we employ an agent that analyze an abstract environment. Such environment also comprises other agents, moreover, the agents interact between them using the decision-making skill provided by the BCA. Analogically, the agents act like a group of persons discussing a problem and trying to built the best solution using their individual point of view. In other words the agents could act, for instance, as specialist with different perspectives and each one provides an individual part of the full solution, consequently, through the interaction or discussion process each member of the specialist group builds a full solution employing the data gathered from the other members. Finally each one of them exposes their full solution proposal and makes in group the final choice about the most suitable solution. Abstracting for the human social behavior, such collective behavior emerges from simple interactions using as cognitive skill a Bayesian cognition approach and organizing the different perspectives using the inference probabilistic results as benchmark.

Part III

Approach

Chapter 4

Proposed approach overview

We have presented the basis and motivations about the main thesis contribution: the matching engine. We have presented it first in a general way to clearly present our starting point. We make evident that there are similar AI approaches but with different goals. Also we funded our approach considering the two main roles needed for an intelligent system: (1) manager agents (2) specialized agents. In our case we propose to employ manager agents as representative of the specialized agents. Also we call them specialized entities instead of agents.

We also consider the specific needs for our approach: using as problem to be solved with software an application specification and to build the solution using MAS meta-models (vowel's approach) as specialized entities (blocks to build the solution).

The present thesis mainly proposes a matching engine to implement the above described approach. However to reach such a goal is needed to solve some satellite situations around it. It means that we need to characterize the application specification data and the meta-models features in order to make it readable to the matching engine. In the next chapter we present narrowly the proposed phases to solve such satellite problems and deeply we explain how is built the matching engine.

4.1 Overview

To our opinion the analysis of multi-agent systems is impossible as unified approach that is using the unified models. We argue that a classification of models and approaches can be made seeking to establish a mechanism of meta-analysis defining a meta-process and meta-knowledge that can guide the designer in choosing the best models tailored that will lead to a precise integration method of architecture (by the identification of appropriate methods).
We understand the notion of meta-analysis in the sense in which this is used in the medicine field (Sutton et al., 2000) as we have stated in section 3.1.1.Therefore for us, the matter is to assimilate "clinical studies" with "problem-domain studies", reasoning over meta-knowledge derived from meta-analysis of our experience and literature to find appropriate solutions in terms of meta-models. We consider that this approach is original in every point in the phase of analysis.

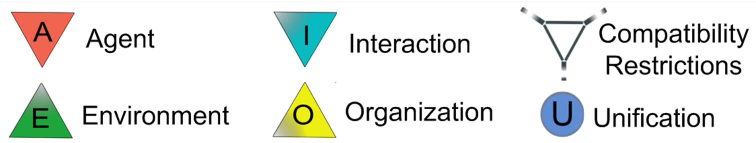


Figure 4.1: AEIO Nomenclature of multi-agent systems

We propose to study the exploring analysis techniques used in the meta-analysis process as occurs in medicine (Sutton et al., 2000) combined with uncertainty treatment (Bessière et al., 2008) and approximate text analysis techniques (Mercier and Beigbeder, 2005), statistical methods (Koehn, 2009), fuzzy, and so on. The use of the concept of micro-arrays (Zhang et al., 2009) associated with the emergent activity seems to be a promising track to operationalize the meta-analysis.

4.1.1 Contributions

We propose two main contributions in the overall project (METALISM introduced in section I):

- The employ of meta-analysis to characterize the meta-meta-knowledge[1] and a conceptual analysis phase process of the multi-agent application life cycle. It will help to better identify applications (Domains + Issues) for which the MAS is also well suited and to provide the characterization tools (perhaps using an ontology) for domains and problems as well as a tie or the comparison (matching). This work is situated within the frame of the Model-Driven Engineering to make it a Model-Driven Engineering for domain and problem.

- Production of a support tool for multi-agent conceptual analysis phase. Based on previous characterization, system modeling for this phase appears to be a good support to build an intelligent system capable of supporting the analysis phase of the design of multi-agent systems.

The present thesis contributes to the project primarily on the production of a global base and path to follow for the entire project. Nevertheless the main contribution consists in the creation of the matching central component (See fig. 4.2) and therefore creating a first prototype for the support tool.

Decentralizing knowledge and experience of the designer in a system composed by multi-agent models and the agentified processes themselves can lead to partial automatization of the analysis phase by using the emergence to produce the best decomposition and make the choice of models. Finally, the capitalization of experimental knowledge (the mental process that the human designer uses to enrich his analysis of the approaches to the problems) will be introduced in this multi-agent tool to assist the designer. In the sequel the mechanisms of meta-learning can be used to enrich the automatic multi-agent analysis tool.

[1]Individual meta-meta-knowledge for an intelligent entity, as a person, organization, society, agent, etc., is represented by the knowledge of knowledge management.

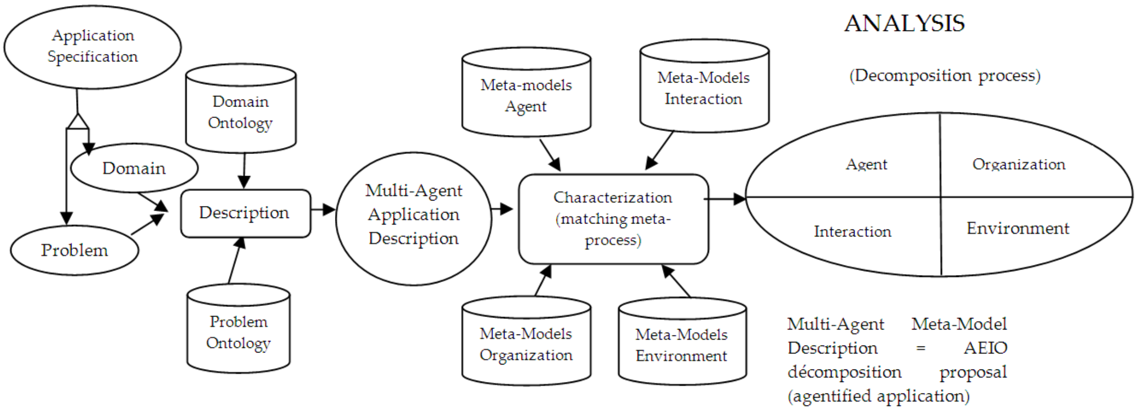


Figure 4.2: Conceptual analysis phases

4.1.2 Activities

It focuses on the conceptual analysis phase of the multi-agent design life cycle, and splits up this phase into two activities (fig. 4.2):

- Activity of description: we must locate here the requirements relative to the specification of knowledge we can collect about the domains and the problems. The result of the stage of description consists in indicating if a multi-agent approach is relevant and if yes to prepare the second stage by enumerating the characteristics of the field and the problem of the application.
- Activity of characterization: the work consists in establishing relations between meta-models (AEIO) and fields and problems characteristics. This stage aims at choosing the properties of the best meta-models which can be suggested for the target application. It is necessary to lay down rules of correlation between descriptions and correspondences of meta-models (meta-knowledge) as well as mechanisms to put in correspondence the descriptions and the meta models (mechanisms of meta-analysis)

4.1.3 Phases

We decomposed the conceptual analysis phase into two main phases which respectively concern the definition knowledge and properties about domain problems and multi-agent models, the study and the tool realization for the matching between application requirements and multi-agent models characteristics:

- Phase A: definition of knowledge and properties about domain/problems and multi-agent models characteristics: The census of the types of application and their distribution on the "field" and "problem" axes will be then approached. One will base oneself on the literature. Usually multi-agent systems are built by integrating agent, interaction, organization, and environment models and by making them operational through the instantiation of these models.

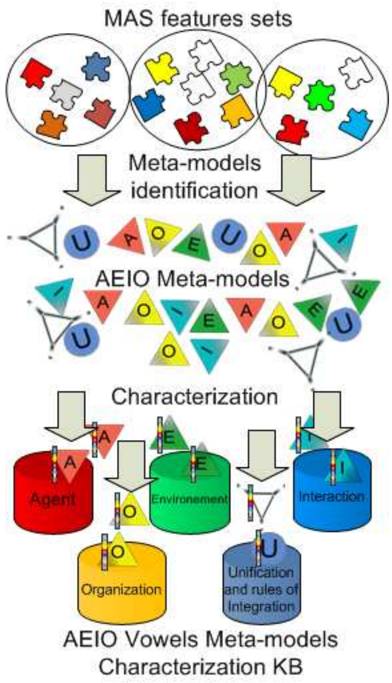


Figure 4.3: Characterization and accommodation AEIO Meta-models within the ontologies

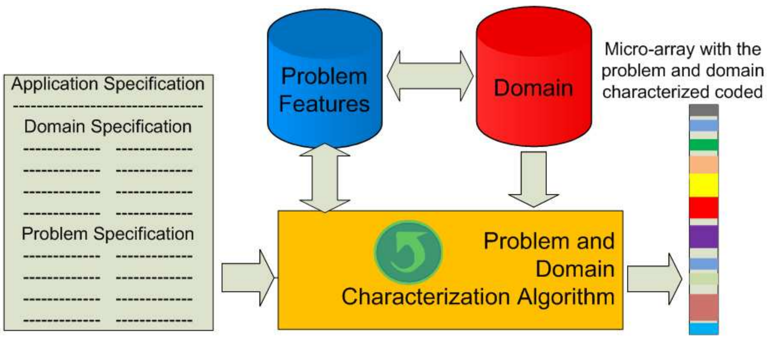


Figure 4.4: Analysis of the problem specification and domain

We will draw up a list of the types of existing models (Agent, Interaction, Organization, and Environment). Classically, existing classification objectives are to compare or to evaluate some models. One will be able to consider classifications of models of the literature, to establish criteria and then to propose a classification. Our classification will lay on the intrinsic characteristics of the models. We will establish relations of operational compatibilities between models, taking into account that the types of models are not all compatible (See fig. 4.3).

- Phase B : Study of the Meta-Analysis process and specification and realization of the tool for the matching between application requirements and multi-agent models characteristics Starting from a census of applications we will establish relations between AEIO and fields and problems. One will study the definition or the adoption of a language to describe the domain and the problem using meta-knowledge. We will seek to build an ontology of domain/problem and an ontology of goal for the SMA to realize. The relations will then be expressed by sizes and criteria to be characterized. According to these values, field and problem of the application to be conceived, it will be necessary to find the good models to be used. One will apply that within the framework of Model-Driven Engineering to make it evolve toward an engineering directed by the models and by domain and problem.

Having a monolithic approach for such a problem is too complex, because of the nature of knowledge and of the properties involved in the process. The proposed system will be able to reason about the established rules of matching about semantic descriptions of the application requirements and of the available models but will be too able to base its process about its previous experience through a reasoning about previous encountered cases. A meta-learning process will be developed.

In order to lay the groundwork for the METALISM project this thesis proposes a development process composed by three different stages in the preliminary analysis phase (See the figures 4.4, 4.5, 4.6 where the nomenclature of such figures is defined in the fig. 4.1). The three stages are:

- Analysis of the statistical specification of the text. The result is a set of likely problem domain characteristics of a micro-coded arrangement such as that used in (Zhang and David, 2005) (see fig. 4.4)
- Matching Engine. Within this stage occurs comparing micro arrays of the problem domain and meta-models AEIO. It is assumed that meta-models AEIO are obtained from the different distinctive features of multi-agent systems (see fig. 4.5) [Occello02]
- Meta-analysis. This stage is constituted by the analysis of previous test results. Among the combinations AEIO meta-models, results of the comparison algorithm, we will seek the most suitable for the given problem. (See fig. 4.6)

The overall result of the analysis phase is an application agentified described in a meta-description of multi-agent system composed of the meta-model AEIO. This meta-description describes how to build each meta-model in the design stage. We propose to make the approach (fig. 4) operational into a multi-agent system which will become a tool of assistance to the analysis. We will propose a specification and a demonstrator to show the feasibility (Task 4.).

4.2 Proposal: MAS Software Engineering previous phase

As we stated previously in the section I this thesis treats the problem of choice the best entities to build a MAS-based solution choosing between many methodologies, many components and many MAS possible solutions. Considering it, our approach proposes a process based on the process described above and visually described in the figure 6. However the process details require to link some context previously defined in 2.4 and 3 with the new concepts about the society of agents described in the previous sections.

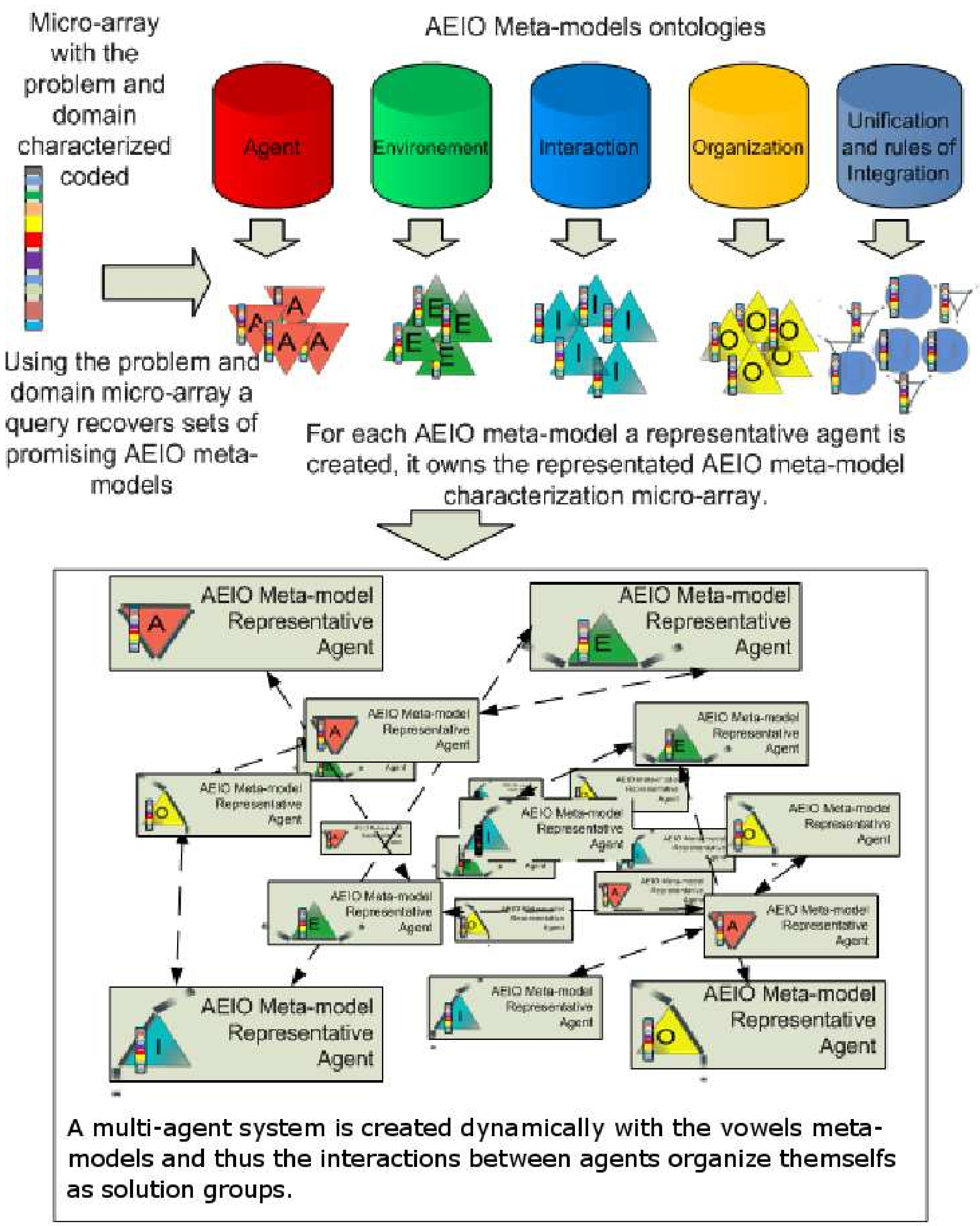


Figure 4.5: Comparison algorithm based on a multi-agent collaborative and emergent behavior

Therefore to our particular case we propose to employ intelligent agents acting as managers of specialized entities and solution builders. In order to achieve such tasks each agent must employ an approach to evaluate the efficacy of their specialized entity in two cases: (1) match the features

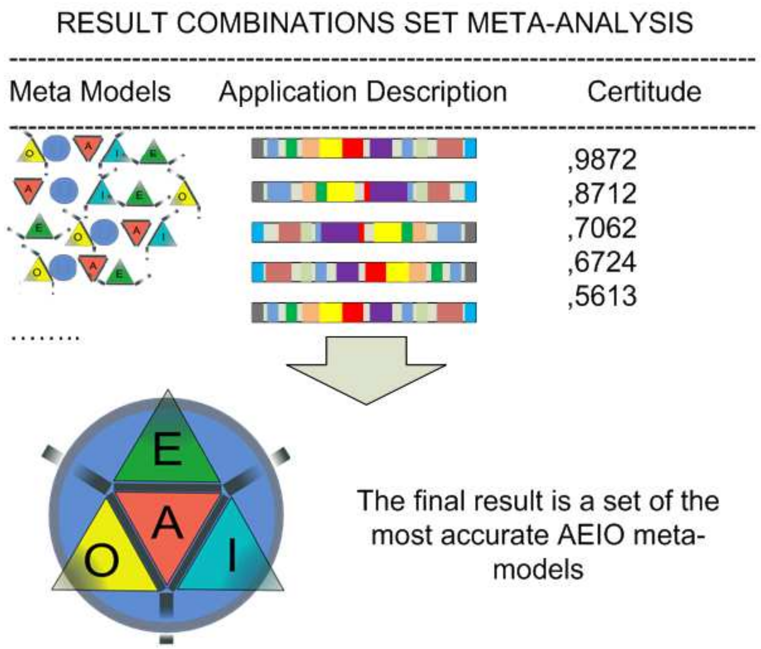


Figure 4.6: Meta-analysis of combinations of sets of meta-models AEIO

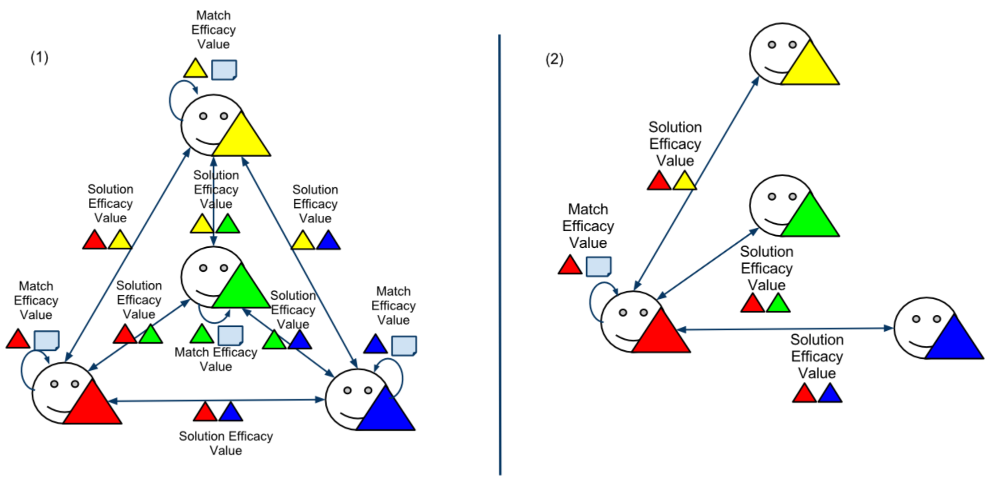


Figure 4.7: Intelligent Agents Solution Group representation with the individual evaluations according in (1) the compatibility of each pair of specialized entities and each entity with the problem description and in (2) with the compatibility from the point of view of one entity with all the other members in a solution group

of their specialized entity with the characteristics of the problem to solve (2) match the features of their specialized entity with the features of other (external and different type) compatible specialized entity. Therefore each agent builds a solution group from the point of view of their specialized entity. Therefore such solution group, partially of fully created, has a set of efficacy results obtained through the match operation between each pair of specialized entities and individually with the characteristics of the problem to solve (fig. 4.7).

Those matching operations require an approach to evaluate compatibilities. An approach permitting to find relations between features and characteristics, and compatibilities between specialized entities. It must be done analogous to the way a doctor recognizes that a patient has disease through the symptoms and then proposes a treatment consisting of several drugs and rules to follow to recover health, where the doctor knows which combination of drugs can be taken at once and certain frequency.

Actually in the field of medicine there is a method that uses the analysis of several studies on the use of one or more drugs on the same disease for clues for finding what is the best treatment for a patient that experiences the symptoms. This method is the meta-analysis. In this thesis we propose to use it in decision-making tasks performed by each agent whereas in our specific case each agent will act as a doctor does when performing a meta-analysis in medicine.

The next step is to apply a meta-analysis method in our context employing specialized entities (MAS Meta-models) and problem to solve: (application specification). It is therefore necessary to create a link between what is meta-analysis and MAS software engineering. Such link is described in the following sections.

Chapter 5

Preliminary Considerations

Before going into deep details of the thesis is necessary to understand some concepts that are not the main objective but are necessary to understand the central part of the thesis. These details are preliminary concepts and definitions and a couple of phases satellites that provide the necessary data inputs for the operation of the main proposal. When it comes to concepts you need to know how they are structured knowledge bases and their relationship to the concept of ontology. Note that this thesis aspires to focus on ontology but simply use a basic form of the concept. Besides introducing the definitions of micro-array, problem domain characteristics and features of Meta-models, the latter including a theoretical way. The satellite phases are composed of process that provides the characterized data inputs using micro-arrays as machine readable standard. Such processes are described below.

5.1 Preliminary concepts

Considering that the present thesis approach requires two sources of information[1] that will be employed in a AI context we have considered to define an ontology way to store such data including a relationship with different domains.

5.1.1 Ontology and Knowledge base

The situation where we need to employ ontology knowledge bases are:

1. Problem characterization.
    - Problem characteristics.
    - Domains.
2. Specialized entities.

[1] defined generally as problem characteristics, domain and specialized entities features.

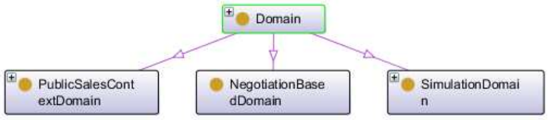


Figure 5.1: Domains Ontology

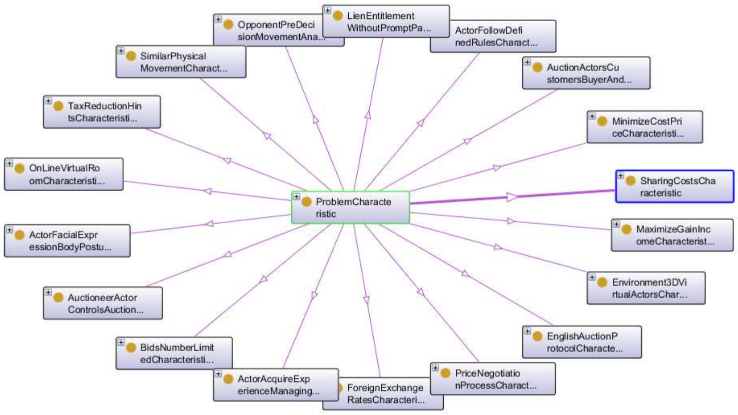


Figure 5.2: Problem Characteristics Ontology

- Meta-models.
- Meta-models’ features.
- Domains.

5.1.2 Ontologies

As word "‘Ontology"’ comes from the Greek term *ontos*, it means "‘being"’, and *logos*, that means "‘word"’. Philosophically ontology refers to the subject of existence. We can say that an ontology studies the categories that exist or could exist within a domain. A domain ontology defines the types of elements that exist within it.

According to (Hendler, 2001) the ontologies in computer science are defined as a set of elements, which have a vocabulary and have connections between elements and rules of inference and logic for a particular purpose. Such definition among others is important for us.

In (Chandrasekaran et al., 1999) from the point of view of AI an ontology means one of two related definitions :

- It’s a representation vocabulary, that usually is specialized within domain or subject matter.

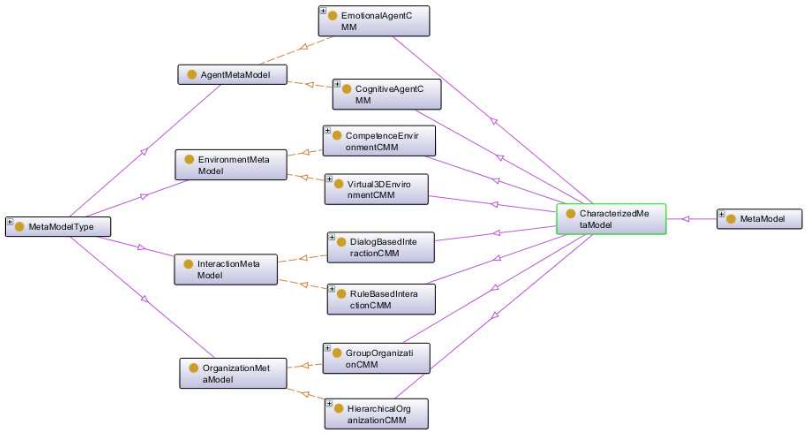


Figure 5.3: Characterized Meta-models ontology including their relationship with vowel's MAS approach

- It's a body of knowledge that describes a particular domain and uses a representation vocabulary.

However both needs an associated underlying data structure to represent the ontology.

Also the most accepted term in AI introduced by (Gruber, 1993) says:

> Ontology is a specification of a conceptualization.

In other words an ontology specifies an abstraction or simplified view of the world.

5.1.3 Meta-models and ontologies

Meta means one level higher of description, thus meta-model is model about model, in other words a model that describes another model. Therefore a meta-model is considered as a characterization to describe other models as an explicit model of the constructions and regulations required to build specific models in a domain of interest.

Considering that such constructions and regulations represent entities in a domain and their relationships, then we consider that this characterizes a meta-model into an ontology. We can take into account, for example, a set of building blocks used to build domain models. Stated another way, a meta-model is like an ontology employed by modelers to build models. For instance, when software developers employs UML to build models of software systems, they actually employs an ontology implemented in such a domain. Therefore this ontology declares concepts such as objects, classes, and relations. However, not all the existing ontologies are created directly as meta-models.

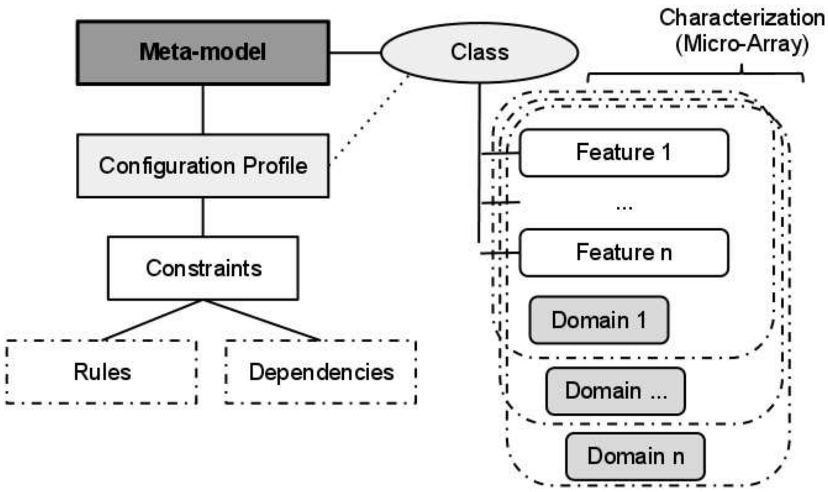


Figure 5.4: Meta-model abstraction with the characterization and configuration profile

5.1.4 Problem Characterization Ontologies

In order to characterize a problem description into a set of characteristics an intelligent system needs to identify them using a set of pre-known characteristics. Therefore such set of pre-known characteristics must be stored into an ontology knowledge base (KB).
For our purposes we need to characterize a particular problem description, this description specifies an application in the abstract is composed of a set of characteristics of problem. To identify these characteristics is necessary to build an ontology containing a semantically structured to accommodate a feature of problem abstractions. Also put in one or more specific domains each feature. Not forgetting that the relationship domain - a feature is defined to pass through a value of efficiency.

We understand a domain as is defined in (Evans, 2003):

> "‘A sphere of knowledge, influence, or activity"’

So, as we mentioned this is the first step and the result is given into a micro-array that contains the information provided by the AS-Characterization process (see section 5.3) and the domain specification (see fig. 5.1). To obtain the micro-array content information we search in a similar manner as is done in (Vijayan Sugumaran, 2002). The difference with our work is that we look into the text from different levels, words, sentences and paragraphs and then we map them into the domain ontology text representation structure to know which the most promising domain is.

Figure 5.9 shows the process of identifying the text-based features. The identification was done by analyzing the text using a technique similar to that defined in (Sánchez and Moreno, 2008) and to create domains using the Web as an information source to create them. In our case we use every description of problem for analysis.
In the section 5.3 we will introduce the context where such characterization will be employed.

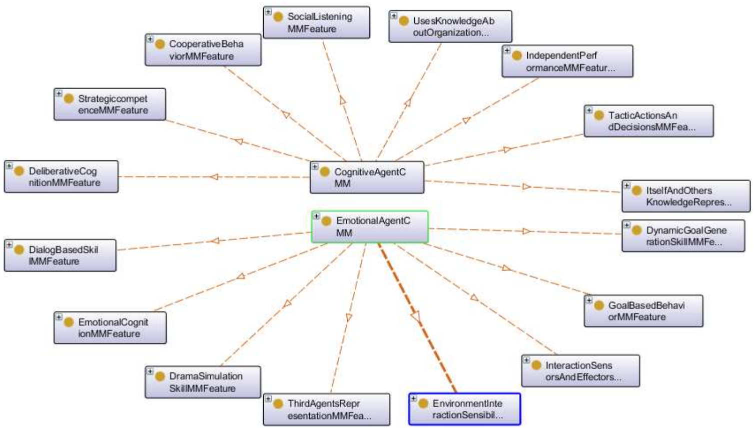


Figure 5.5: Agent Meta-models Features Ontology

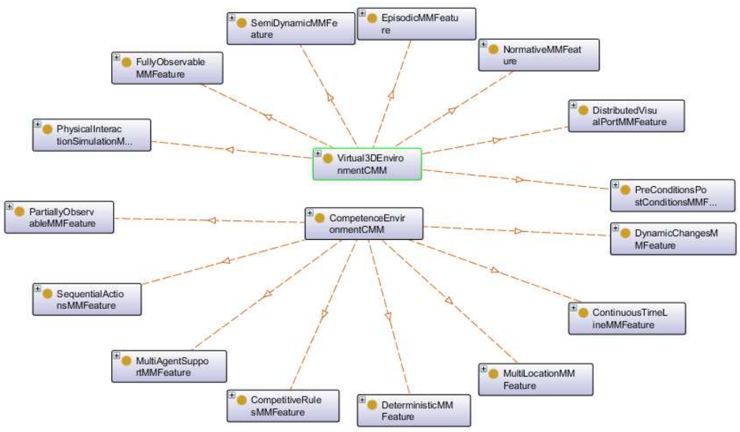


Figure 5.6: Environment Meta-models Features Ontology

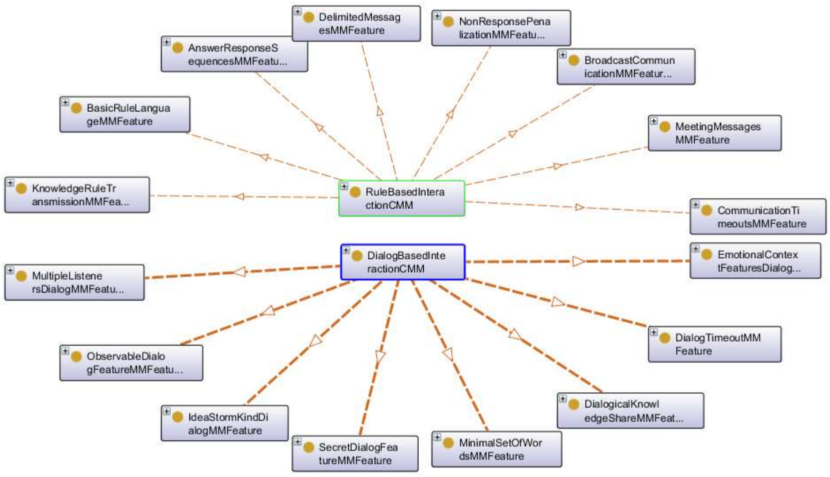


Figure 5.7: Interaction Meta-models Features Ontology

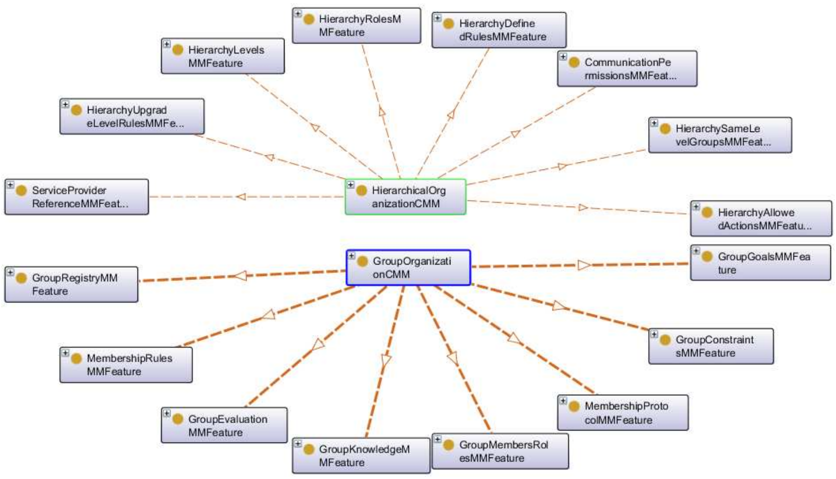


Figure 5.8: Organization Meta-models Features Ontology

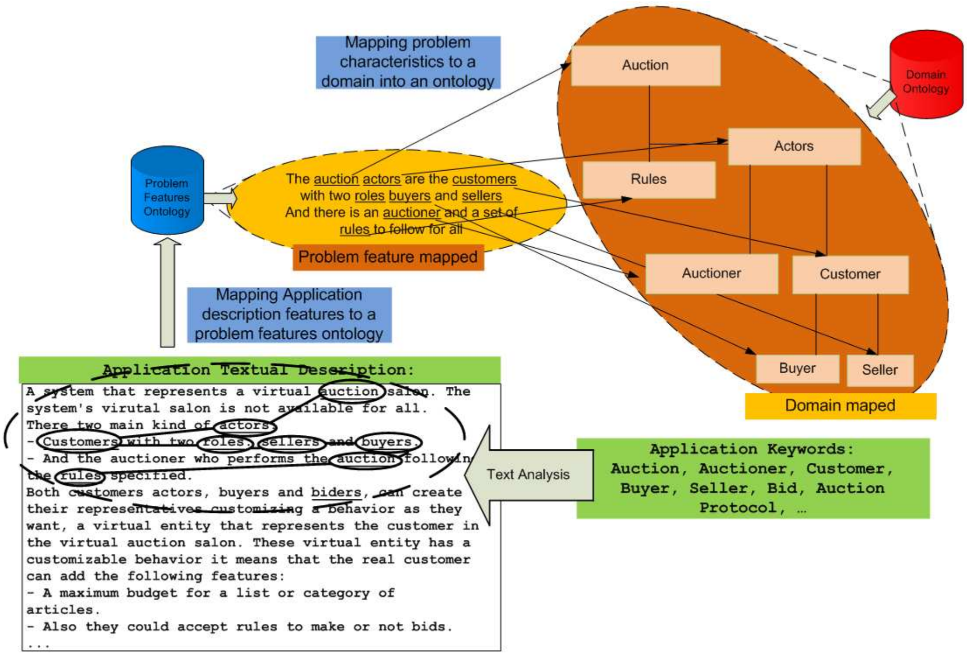


Figure 5.9: Ontology Text-based analysis process graphical overview example

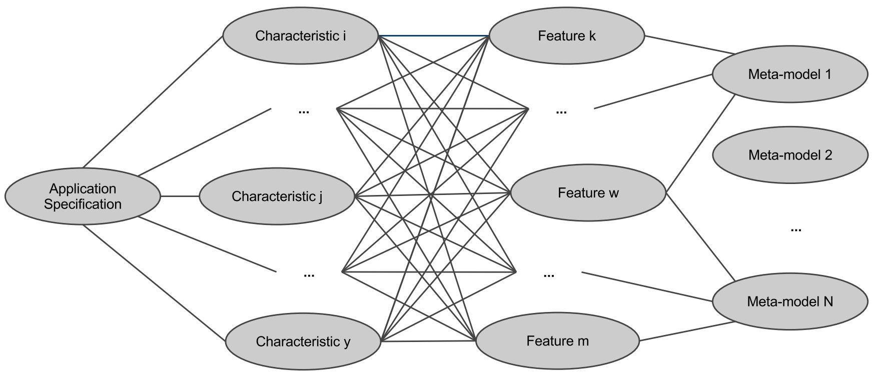


Figure 5.10: Meta-model candidate selection using the AS-Characterization

5.1.5 Meta-models Knowledge base

In our case we propose to employ meta-models created from existing methodologies using the AEIO vowel's approach. Such meta-models are defined using an abstraction method[2], manual or automatic. Nevertheless in our prototype we have employed meta-model features defined by textual sentences to simplify the constructions. Therefore we have defined a meta-model concept as is shown in the fig. 5.3. Such meta-models are built using a set of features, therefore, they are selected as candidates making a efficacy value link between MM-features and AS-characteristics (see fig. 5.10).

It is performed considering that each feature is related to a domain using a efficacy value. In

[2]Such a method is out of the scope of this thesis.

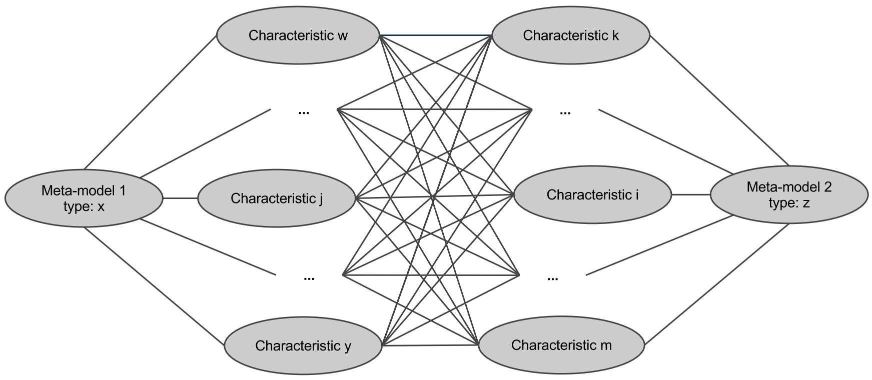


Figure 5.11: Meta-model Unification Memory Abstraction

other words there are several links considering a domain, however, such domain must be selected previously in the AS-Characterization process (see fig. 5.2) Therefore the knowledge base has been declared using a XML standard showed in 6.4.3. In the future work section we discuss improvements to this method.

In order to build solutions employing many meta-models we need to create a unification memory as showed in fig. 5.11. So, we can link each pair of meta-models considering an efficacy value that relates the features of both different meta-models.

5.2 Preliminary phases

Before starting the operational details of the comparison engine is necessary to define several concepts that provide to the engine with the necessary information to make decisions. This section defines concepts as application specification, problem characteristics, domain, application specification characterization, micro-array, meta-model, meta-model type, meta-models characteristics, meta-model characterization.

5.2.1 Micro-array

The micro-array concept comes from the field of bio-informatics especially from the analysis research of deoxyribonucleic acid (DNA)(Zhang and David, 2005) that characterizes and find genetic patterns. Such a micro-array is composed by compressed genetic information that stores a complete profile in a small amount of data. In our case we employ micro-arrays that represent an application specification characterization within a solution domain and also MAS meta-models characterizations.

5.2.2 Application specification characterization

Commonly the system designer passes first through the requirements gathering step to obtain the application specification (AS). To see if the desired application is possible to solve using a MAS-based solution is required to identify the AS problem characteristics and domain. To do this it is required to have a good background in MAS also to be related with the domain and problem. However generally the engineers have problems to get connected with the domain of the application that they must create (Evans, 2003). Therefore to choose the most accurate MAS methodology and use it to develop a MAS-based solution requires having a good MAS methodologies knowledge and MAS paradigm background. These issues are some of the main lacks related to the decision-making area (Dastani et al., 2004). For these reasons we propose to characterize the application specification as the user input. This must be done through a process that recognizes the characteristics and found the most suitable domain for the entire set of characteristics. Even so such process is not a goal of this thesis.

5.2.2.1 Domains and AS-characteristics Knowledge Base (KB)

First we define the known abstract domains and AS-characteristics sets that are available in the knowledge base (KB). For us a domain abstraction is represented as δ_k. Moreover we have a number d of domains abstractions stored on the KB set D_{KB}:

$$D_{KB} \equiv [\delta_1, \ldots, \delta_d] : \forall_k \in [1, d]$$

As well, we symbolize an AS-characteristic abstraction as κ_k. In addition we have a number c of known AS-characteristics abstractions stored in a KB set C_{KB}:

$$C_{KB} \equiv [\kappa_1, \ldots, \kappa_c] : \forall_k \in [1, c]$$

Therefore the AS-characteristics abstraction KB is built using the next matrix structure:

$$AS_{KB} \equiv \begin{bmatrix} \theta_{1,1} & \ldots & \theta_{1,c} \\ \vdots & \ddots & \vdots \\ \theta_{d,1} & \ldots & \theta_{d,c} \end{bmatrix}$$

Such matrix structure is assembled with real values from the interval $[0, 1]$. These values relate the domains δ_i with the AS-characteristics κ_j. Therefore such values are represented as $\theta_{i,j}$ and stands for an efficacy percentage of the AS-characteristic κ_j in the domain δ_i.

5.2.2.2 AS-characterization

Each AS passes through an AS-characteristics recognition process that matches the AS-characteristics of the KB. Such a process determines the known AS-characteristics that best define the AS analyzed. The result of this process is stored in a micro-array [3] that stores a codified representation of

[3] The micro-array is a codified data representation - usually genetic - similar to the one employed in (Zhang and David, 2005)

the AS-characterization. In view of the characterization process an individual AS-characterization micro-array uses only a subset of the existing characteristics and domains stored in the KB. Thus, we define C' and D' as a subset of C_{KB} and D_{KB} respectively: $C' \subseteq C_{KB}$ and $D' \subseteq D_{KB}$. Therefore we define the cardinality of each set $|C'| = c'$ and $|D'| = d'$, in other words, the number of characteristics and domains selected are c' and d' respectively. Considering that we have characterized any AS_k - through the AS-characterization process - we define the single AS-characterization micro-array in the next manner:

$$AS'_k \equiv \begin{bmatrix} \theta_{1,1} & \dots & \theta_{1,c'} \\ \vdots & \ddots & \vdots \\ \theta_{d',1} & \dots & \theta_{d',c'} \end{bmatrix}$$

Where the number of columns - represented as c' - is the total number of AS-characteristics matched in the AS-characterization process. Moreover the number of rows - represented as d' - is the total number of domains related with the found AS-characteristics. Thus $AS' \subseteq AS_{KB}$ where AS' denotes the resulting AS-characterization micro-array.

5.2.3 Meta-models Characterization

Considering the meta-analysis nature we propose to characterize these meta-models and evaluate each meta-model's features considering a link of each feature with a domain of solution. That means that each characterized meta-model has a set of features, each feature is related with different domains of solution using efficacy values as relation. These values denote the feature efficacy within each different domain. This characterization allows fully assessing the meta-model efficacy within a domain. Therefore you can also evaluate the effectiveness of a set of meta-models. However, to obtain these efficacy values, we need skilled engineers to enter these values manually or automatically create a knowledge acquisition process, but, these problems are outside the focus of this thesis.

5.2.3.1 Meta-models types and features

To define the meta-models abstractions KB we state first that there are four different types of meta-model (MM) in the following KB sets:

$$A_{KB} \equiv \{\alpha_1, \dots, \alpha_a\} \forall_k \in [1,a] \qquad E_{KB} \equiv \{\varepsilon_1, \dots, \varepsilon_e\} \ \forall_k \in [1,e]$$

$$I_{KB} \ \equiv \{\iota_1, \dots, \iota_i\} \ \forall_k \in [1,i] \qquad O_{KB} \equiv \{o_1, \dots, o_o\} \ \forall_k \in [1,o]$$

Where each KB set stores a different MM-type: Agent, Environment, Interaction and Organization respectively. Each any MM α_k, ε_k, ι_k and o_k are different MM-types between them. Also in each type there are different kinds of MMs, for example, we can have two different MMs of type agent as α_1 representing a "cognitive agent", and α_2 representing a "reactive agent" and ε_0 as "observable environment" and ε_1 as "3D virtual environment".

Generalizing the four MMs abstractions KB sets we define the KB super set T_{KB} as a KB super set containing all the known MM-types in the next manner:

$$T_{KB} \equiv \{\tau_1, \ldots, \tau_t\} \forall_k \in [1,t] \qquad T_{KB} \equiv \{A_{KB} \wedge E_{KB} \wedge I_{KB} \wedge O_{KB}\}$$

Where each τ_k is a MM-abstraction of any MM-type. This KB super set generalization is useful for the following definition of MM-characterization.We define the set F_{KB} as the KB set of MM known features abstraction in the next form:

$$F_{KB} \equiv [\varphi_1, \ldots, \varphi_f] : \forall_k \in [1, f]$$

Thus we consider each φ_k as a MM-feature abstraction of any MM-type.

5.2.3.2 MM-Characterization

In order to create a MM-characterization KB each MM must pass through a MM-features recognition process - similar to the AS-Characterization process described in 5.2.2.2 but applied to MM-features against a domain - that determines which known MM-features best describes each MM. However the MM-Characterization process is out of the present thesis goals but a minimal approach is addressed in the Appendix B. Considering that we already have characterized any MM_{τ_k} we state the MM-characterization result as the next matrix structure:

$$MM'_{\tau_k} \equiv \begin{bmatrix} \mu_{1,1} & \cdots & \mu_{1,f'} \\ \vdots & \ddots & \vdots \\ \mu_{d',1} & \cdots & \mu_{d',f'} \end{bmatrix}$$

Such matrix structure is built with real values represented as $\mu_{i,j}$. Such real values are from the interval $[0,1]$. The number of columns f' is the number of matched features and the number of rows d' is the number of domains related to the MM-features found. Moreover, each $\mu_{i,j}$ is an efficacy value that relates each MM-feature φ_j found with each related domain δ_i.

Consequently the KB set composed by all the MM-Characterizations and features has the next structure:

$$MM'_{KB} \equiv \begin{bmatrix} MM'_{\tau_1} & \ldots & MM'_{\tau_n} \end{bmatrix}$$

5.2.3.3 Discriminating factor

The previous KB structures of MM show that they are organized by MM-types nevertheless the domains is the major discriminating factor. Each MM-type contains a matrix where the rows are the domains and the columns are the features. Therefore choosing a small number of domains allows to select only the rows related with such domains, therefore, use only the efficacy values $\mu_{i,j}$ related with each features within these domains. Finally, each MM uses the same definition of domains employed in the section 5.2.2.1 of this thesis.

5.2.3.4 Unification rules knowledge base

The unification rules are built as an hyper-matrix of features φ_k relations. Where each feature is related with each other for each different MM-types. For an individual two MM-types is as follows:

$$u_{\tau_i,\tau_j} = \begin{bmatrix} & & & \tau_i & \\ & & \varphi_k & \cdots & \varphi_l \\ & \varphi_n & \begin{bmatrix} \upsilon_{1,1} & \cdots & \upsilon_{1,m} \\ \vdots & \ddots & \vdots \\ \upsilon_{m,1} & \cdots & \upsilon_{m,m} \end{bmatrix} \\ \tau_j & \vdots & \\ & \varphi_p & \end{bmatrix}$$

Moreover, this structure works as a memory that relates each MM-type's features with each other adding a compatibility value. Such values are obtained through an automated solution to learn it or if needed is possible to write them manually - based on the experience of the developer -. However this thesis does not focus on this but uses it for the main thesis purpose. We fulfilled these values and the MM-KB using a survey answered by the experienced laboratories members in the area (also stated in the subsection 5.2 and 6.5.4.2).

5.3 Satellite phases

The process to characterize an AS is out of the main scope of this thesis, nevertheless, in order to feed the matching engine (the main phase) we need to define a minimal approach to characterize a text-based AS. Therefore in this chapter we present a basis of preliminary definitions needed for such characterization approach and for the matching engine and we introduce the implementation of a sub-prototype to perform such characterization. Moreover the characterization approach is described in the next section, however, we state that it is only a shallow approach that help us to build up the path to arrive to the matching engine.

5.3.1 Application Specification Text-based Characterization

In the software engineering context finding the features that best describe a problem regularly leads us to realize which the problem's domain is. The recognition of problem's features and domains gives us enough clues to plan the construction of the problem's solutions. However, finding these features and problem domains requires the developer to have experience, a close relationship with the domain and knowledge of it. Usually these facts make difficult to perform a correct characterization of a given problem for the not linked to domain and inexperienced engineers. This appendix presents a minimal approach that proposes the use of a Multi-agent system (MAS) based engine for analyze the textual application description and obtains their problem's features and domain characterization for a possible MAS-based solution. This engine comprises MAS architecture and pattern-based text recognition. It also proposes the use of a semantically structured knowledge base of problem's features and domain's specifications linked to the text pattern results. Furthermore our approach main goal is to act as an dynamic and upgradeable assisting approach for the engineers in the characterization of a problem using a description of the desired application.

A great variety of multi-agent methodologies can be found today allowing the software development conduction by means of multi-agent models (for a MAS survey (Federico Bergenti, 2004)). On the other hand, software designers are indecisive to use them since choosing one method that could carry out to attach the type of models involved. In consequence when the designers are choosing a method they must master the multi-agent model properties and also their must know how to match them to the application's specifications and domain. In circumstances of software production a common problem for the engineers is the application's domain misunderstanding (Evans, 2003) because the engineers are not familiar with all the domains where they have to work for a solution; usually they must identify the domain and consequently learn the domain's background to manage with the development process. Thus, the learning curve for acquire the required skills is steep. We think this is one of the reasons restricting the dissemination of multi-agent methods. The objective of this satellite phase is to propose a minimal process approach to identify characteristics from a textual description of an application specification. The resulting data of such analaysis will be employed to feed the matching engine described deeply in this thesis.

5.3.1.1 Related work

There are many approaches that aspire to define a path to abstract information from raw data. In the approach defined by (Sánchez and Moreno, 2008) there is a process that allows to create domains from web-based raw information and using wordnet (Miller, 1995). Wordnet is considered the most reliable and employed on-line lexical and semantic repository for the English language. It declares a lexicon, a thesaurus and semantic links with the English terms. It classifies words into categories and relates their meanings. Therefore We propose to organizes such text structures in order to connect them with a problem characteristic that could be identified within a application specification as a textual problem description.

We also consider that our approach is like a translator that takes an textual description to translate it into a format machine readable. Therefore we have found statistical machine translation techniques (Koehn, 2009) that are relevant to our proposed approach. Such a kind of techniques are employed also by Google Translator (Google, 2011) gathering statistical information globally to improve the translation software process. Therefore we propose to create a complementary MAS architecture to make possible to distribute the AS-characterization phase in order to gather statistical data easily similar as Google does.

We take into account such domain generation process and statistical translation in order to create a domain to be employed as basis to identify such word structures related to problem features within a text that contains an application specification.

5.3.1.2 Complementary MAS Overview

The present satellite phase proposes to create a complementary MAS-based phase to analyze a textual application description with the objective of scrutinize the text and found information to identify their problem's characteristics and solution domain. Such data is useful to locate the Meta-models candidates that are related (known as good to solve) to each of the problems characteristics. So, this satellite phase is composed by an artificial intelligent analysis process by means of a MAS approach. This process typifies each application description into a characterization assay that we call application description micro-array characterization assay (ADMACA). Each ADMACA is composed by the statistical data outcome about the identification of problem's features and domain's specification. This satellite phase also defines a XML-based standard to manage, share, reuse and process the ADMACA results of each analysis. Basically our goal is to provide an approach to support the engineers through the problem characterization process in order to simplify the identification of the application description problem's characteristics and specific domain. Similar as occurs in every meta-analysis this enables to create a kind of protocol of data acquisition, in our case text-based analysis, employing a standard to make the gathered data reusable.

Finally the present characterization process complements the matching engine defined at Section 6.3 feeding it with the micro-array characterization, as a machine readable input, that allows such an engine to perform is matching process. Remembering that the matching process considers also the use of MAS meta-models[4] characterizations.

5.3.1.3 Complementary MAS Architecture

The MAS architecture proposed comprises a set of agents with different task and behaviors. The architecture is divided in two main fields: the MAS and the web-services; as we can see in the example of the fig. 5.13. The MAS field is composed by four agents, where each agent performs different task that are combined to make emerge as result the application description (AD) characterization. Each characterization is considered as an assay that corresponds to a specific application. The characterization is composed by patterns references to design elements of characterization as problem features and domain specifications and quantification parameters as belonging values. These

[4] A meta-model is a model description or model of models in our case we consider the use of meta-models inspired in the MAS approach AEIO (Demazeau, 2001) that decompose a MAS in four components Agent, Environment, Interaction and Organization that will allow to match these components with a characterization assay. Nevertheless this appendix approach only considers the characterization of an application description process.

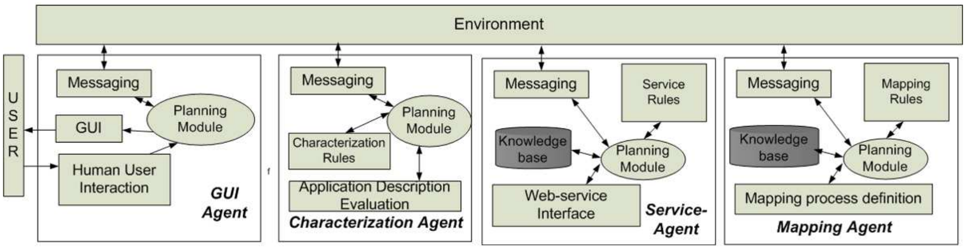


Figure 5.12: MAS for characterization architecture overall.

data must be recovered through the characterization process using the information provided by the agent's tasks. So we describe briefly each agent tasks in the following lines:

- GUI Agent. Their main task is to get in touch with the user and it receives the user input and manage it with the system responses to the user. In our case we have defined as user input the application description text. This agent manages the input content and retrieves questions or results to the user if the input contains insufficient information it ask for more data. So, it means that this agent can contact to the user to ask for more information if it is required. Thus the constitution of the GUI Agent comprises a Graphic User Interface, A semi-dialog-based human user interface with user Input-output management, and finally a message based behavior that allow receive messages from the rest of the agents and translate them into human understandable information.

- Characterization Agent. Its main task is to manage the characterization process, so we must define a process to characterize the user input. Therefore this agent owns a cognitive behavior that manages and plans the characterization process.

- Service agent The main task is to manage a web service connection. Thus, this agent receives request from the characterization agent asking to perform some tasks related to the characterization activity. It works as an agent that manages a web-service in a transparent way to provide the service as an agent. So, it allows improving or changing the web-service and only modifying the agent that provides this service.

- Mapping agent This agent performs the identification and mapping of data (for example text patterns) received from the characterization agent. It is done through a cognitive decision making that matches the data with the existing data in the knowledge base, that could be defined as we propose in a problem features ontology. In our case we propose that the pertinent information about the problem features and domain specification ontologies is recovered through the Ontology Connection Agent and stored temporally in the knowledge base of the Problem Features Detector Agent and the Domain Locator Agent, both are considered similar to this kind of agent.

- Using our MAS-based characterization approach with text pattern recognition

Using these kind of agents in a characterization process allows us to create a MAS-based solution to characterize an application description and other kind of similar characterization task. In this appendix approach we propose as example the use of text in the application description and we propose to use text pattern recognition to characterize the data. Thus particularly we propose the implementation of the next agents:

- Features Detector Agent. This agent performs the identification of problem features using the text patterns received from the text analysis agent. It is done through a cognitive decision making that matches the text patterns with the existing features defined in the problem features ontology. The information about problem features ontology is recovered through the Ontology Connection Agent.
- Domain Locator Agent. This agent locates the possible domain of the application description using the text structure patterns, the problem features related with the patterns and the existing domains through the Ontology Connection Agent. So this agent performs a main cognitive task to induce the possible domain or domains for the text structure patterns received from the Text Analysis Agent.
- Perl Agent. The main task is to manage the Perl scripting web service connection. Thus, this agent receives request from the Text Analysis Agent asking to parse a complete or partially a text. Also this agent manages the regular expression (regex) rules employed in the process. It works as an agent that manages a web-service in a transparent way to provide the service as an agent. So, it allows improving or changing the web-service and only modifying the agent that provides this service.
- Ontology Connection Agent. The main task is to manage the ontology connection web service. So this agent receives request from the Features Detector and Domain Locator agents to query the ontologies (see fig. 5.13). So it works similarly as the Perl Agent giving an agent interface to a web-service.

5.3.1.4 Application Description Text Patterning

With the objective of showing a minimal working example we have defined a basic text recognizing process that we will explain in this section, nevertheless our goal is not to propose a new approach about text recognizing, so we only explain it as complementary approach that works within the AS-characterization process. So we have defined the text recognizing process using a multi-layered analysis to extract the relevant data from the text. In order to characterize the data we propose to divide the complete text document into 3 main-layers or steps of analysis: paragraphs, sentences and words. The goal of this division is to create a text pattern composition similar to a Bayesian network (Stuart Russell, 2009). We have chosen this topology because this divides the data and keeps the nature of the text analysis comprising two ways top-down and bottom-up using Perl scripts inspired

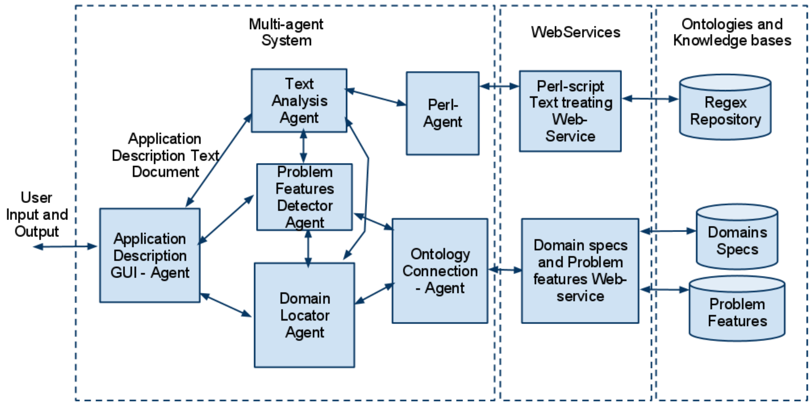


Figure 5.13: Example of a text pattern recognition based characterization.

in the text mining process utilized in (Bilisoly, 2008). Also the Bayesian inference nature is related to the decision making using the evaluation of probabilities values with the statistical and historical information to face uncertainty; this feature is ad hoc for our artificial intelligence approach since we manage with an application description provided by a human user. So, the first step of the text analysis algorithm is go down the text structure starting from the top document level splitting it in paragraphs; then each paragraph branch is separated into sentences; consequently each sentence is divided into words. At this step we have descended into the text structure in a way similar as exposed in fig. 5.14. Formally we define each set of words, sentences and paragraphs as follows:

$$W \equiv \{\omega_1, \ldots, \omega_w\} \, \forall_k \in [1, w]$$

Where each ω_k is a valid word found in the application description. We use subsets of these words to create sentences. Thus the sentence set is composed as follows:

$$S \equiv \{\sigma_1, \ldots, \sigma_s\} \, \forall_k \in [1, s] : \sigma_k \subset W$$

Using the subsets of sentences we build paragraphs that are formally defined into a paragraph set in this manner:

$$G \equiv \{\rho_1, \ldots, \rho_g\} \, \forall_k \in [1, g] : \rho_k \subset S$$

Also we use a set of real values where each one represents a result of the application description text analysis as a ratio value for each related text structure:

$$V \equiv \{\nu_1, \ldots, \nu_v\} \, \forall_k \in [1, v] \, \nu_k \in \Re, [0, 1]$$

Therefore, the second step is to filter the word level leaving only the keywords; including proper names, adjectives and verbs (all variations: conjugated, infinitive, etc) but dismissing pronouns and connectors. Then we give a value to each keyword performing a ratio operation from the resulting

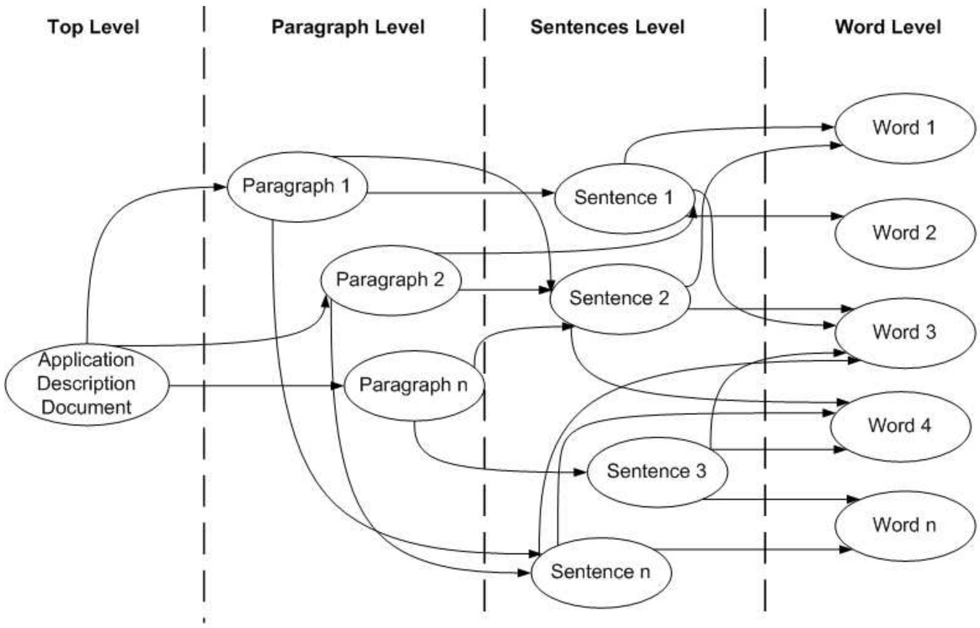


Figure 5.14: The top-down sense in the text pattern recognition process proposed.

keyword subset of the last step and counting each subset's word occurrences [5] in the subset dividing it by the total number of words in the subset. So, this operation is performed by the following formula:

$$\nu_{\omega_k} = \frac{occurrences(\omega_k)}{|W|}$$

Where ν_{ω_k} is the keyword ωk ratio value and $occurrences(\omega_k)$ is the occurrences number of the keyword ω_k that has been found in the application description and $|W|$ is the set of keywords W cardinality, it means the total number of different keywords.

A fragment of the described result is showed in fig. 5.15 where we can see a XML-based standard that defines an example of the keyword patterns found. Each keyword tag contains tags with occurrences, where each occurrence has the paragraph and sentence id where it was found.

The next step ascends through the keywords patterns from the keyword level to the sentence level, using the keywords significance value we assign a new significance value for each sentence. We take each one related to sentence words values to sum all of them and obtain the words-related-to-sentence average; then we take the average as the significance value of the sentence. We repeat this step with each sentence related to a paragraph. The formula to obtain the value of a sentences σ_k is:

$$\nu_{\sigma_k} = \frac{\sum_{i=1}^{n} \nu_{\omega_i}}{n}$$

Where ν_{σ_k} is the significance value of the sentences σk, and n is the total words related to the

[5]We mean occurrences as a word or a variation related with this word, as example words that are in plural or singular are considered as the same word.

```xml
<word value="auction" overall =0.0573248407643312 >
    <occurence id="1" para_id="1" sen_id="2" />
    <occurence id="2" para_id="1" sen_id="4" />
    <occurence id="3" para_id="2" sen_id="6" />
    <occurence id="4" para_id="3" sen_id="10" />
    <occurence id="5" para_id="3" sen_id="11" />
    <occurence id="6" para_id="3" sen_id="11" />
    <occurence id="7" para_id="3" sen_id="11" />
    <occurence id="8" para_id="3" sen_id="11" />
    <occurence id="9" para_id="3" sen_id="11" />
    <occurence id="10" para_id="1" sen_id="4" />
    <occurence id="11" para_id="3" sen_id="10" />
    <occurence id="12" para_id="3" sen_id="11" />
</word>
<word value="automated" overall =0.00636942675159236 >
    <occurence id="1" para_id="3" sen_id="10" />
</word>
<word value="available" overall =0.00636942675159236 >
    <occurence id="1" para_id="1" sen_id="2" />
</word>
<word value="behavior" overall =0.0127388535031847 >
    <occurence id="1" para_id="2" sen_id="6" />
    <occurence id="2" para_id="2" sen_id="7" />
</word>
<word value="biders" overall =0.0127388535031847 >
    <occurence id="1" para_id="1" sen_id="4" />
    <occurence id="2" para_id="2" sen_id="6" />
    <occurence id="3" para_id="2" sen_id="8" />
</word>
```

Figure 5.15: Word patterns middle-result of the text analysis.

sentence σ_k and each $\nu_{\omega_i} \in \Re[0,1]$. And the formula for a paragraph ρ_k is:

$$\nu_{\rho_k} = \frac{\sum_{i=1}^{m} \nu_{\sigma_i}}{m}$$

Where ν_{ρ_k} is the significance value of the paragraph ρk, and m is the total sentences related to the paragraph ρ_k and each $\nu_{\sigma_i} \in \Re[0,1]$.

Finally the result is a text-based multi-layered pattern structure weighted. This structure allows us to explore it starting from any layer. That means we can match similar text structures to see if there are similar sentences or paragraphs using the keywords but evaluating the overall at sentences or paragraph level (See fig. 5.16). This text pattern recognition approach is useful in two main sides:

- First, for automated learning using knowledge engineering it allows storing the text patterns structures with their values linking them to a problem's feature or a domain's specification.

- Second, for automated characterization we can evaluate the incoming application descriptions using the historical results stored using the first part. This will provide us an automated process to characterize problem's features and domain's specifications.

Nevertheless in this appendix approach we only focus on the second side, where the text patterns are found and characterized using the MAS engine. Our objective is to show how we can provide a possible solution by means of MAS to characterize an application description. However it is possible to use a different method that the pattern recognition to text.

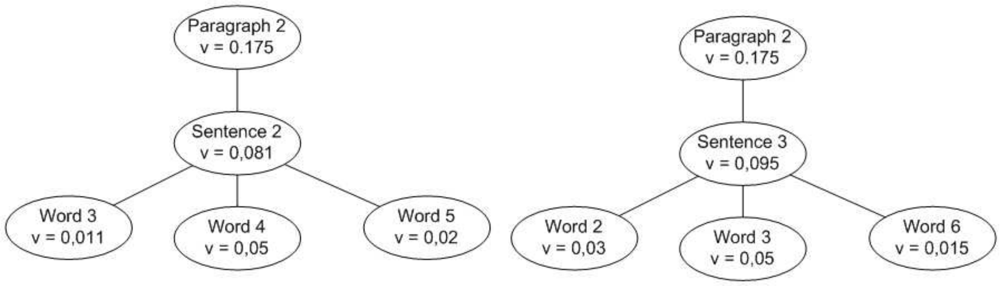


Figure 5.16: A graphical representation of the text patterns with the relations at sentence level.

So, this task is performed by the Perl agent and the result is the text patterns. These patterns are mapped with the problem features and domain ontologies similar patterns. So, this mapping process is graphically showed in the fig 5.17.

5.3.1.5 Application Description Micro-array Characterization (ADMACA) Standard

The use of a standard to register the results provides us a way to store, manage and reuse the assays produced with the described approach. One of the main reasons for creating this standard is to make available the results as a repository of independent but related assays in the characterization context with the objective of leaving the road ready for the arrival of a supplementary process of meta-analysis or mining data. So, we have defined a XML-based standard that comprises the following data sections:

- User profile data. It contains the user personal and contact information, also the institutional information if required. See the example file fragment code below:

  User profile data

```
<profile>
      <developer>
        <fullname>
          <firstname>Michel</firstname>
          <lastname>Occello</lastname>
        </fullname>
        <organization>
          <id>1</id>
          <name>University of Grenoble - Laboratoire LCIS</name>
          <description>Complex Systems Group Development</description>
        </organization>
        <email>
          <username>Michel.Occello</username>
```

```
        <server>iut-valence.fr</server>
      </email>
      <webpage href="http://lcis.grenoble-inp.fr" />
    </developer>
  </profile>
```

(Fragment from a example file wrote with the ADMACA standard)

- Acquisition protocol employed data. This section has the title and description of the protocol used in the characterization; in our case we use the text-pattern analysis. Nevertheless the acquisition protocol could be different that the text-pattern analysis. This section is accompanied by the labeling protocol composition; this is the elements evaluated and the resulting quantification parameters. In our case, the protocol stores the problem features and domain specification as mapped elements with their belonging probabilistic values as quantification parameters. See the next file fragment as example:

  Protocol data and labeling data

```
<protocol>
  <id>1</id>
  <name>Text Pattern Analysis</name>
</protocol>
<labeling id="1" name="Problem-Domain Characterization">
  <designElements>
    <element type="Problem feature">
      <design-id>1</design-id>
      <title>Maximum gain feature</title>
      <description>Optimize the gain finding the lowest
      cost or price
      </description>
    </element>
    <element type="Problem feature">
      <design-id>2</design-id>
      <title>Auction roles feature</title>
      <description>The auction actors are the customers
      with roles of buyers and sellers.
      </description>
    </element>
    <element type="Domain Specification">
      <design-id>3</design-id>
```

```
        <title>Automated auction</title>
        <description>The automated auction is composed by actors,
        where each actor could take the role of auctioneer or customer;
        also a customer can act as buyer or seller.
        </description>
      </element>
    </designElements>
    <quantitationTypes>
      <type>
        <quantitation-id>1</quantitation-id>
        <title>Relevance</title>
        <description>Belonging probability value</description>
      </type>
    </quantitationTypes>
  </labeling>
```

(Fragment from a example file wrote with the ADMACA standard)

- Resulting characterization data. The results of the assay are stored using the id references to the defined elements and acquisition parameters in the labeling protocol. See the next fragment code example:

  Characterization data

```
<characterization id="1">
  <value design-id="1" quantization-id="1">0.7845</value>
  <value design-id="2" quantization-id="1">0.9312</value>
  <value design-id="3" quantization-id="1">0.8923</value>
</characterization>
```

  (Fragment from a example file wrote with the ADMACA standard)

5.3.1.6 Results

The present approach has been tested using some application description text and modifying several times the regular expressions to improve the text pattern recognition method; this task belongs to the Perl Agent working process. We have performed it without modify the entire system. Also we have added and retired manually entries to the problem features and ontology specifications ontologies. Thus these facts show us that the MAS approach implemented in this process is adaptable and

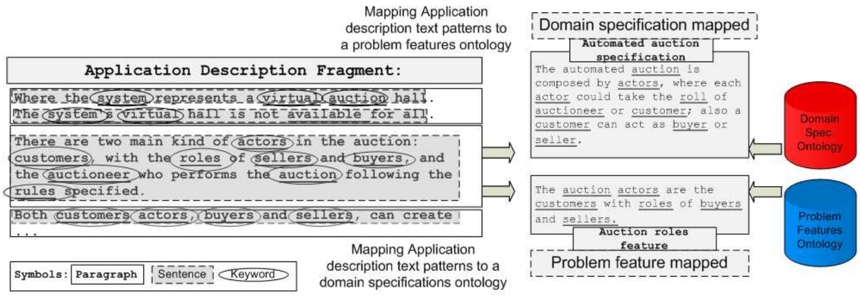


Figure 5.17: Mapping application description text patterns with ontology text patterns.

tolerate the dynamic updates. The characterization assays has been performed as expected. Nevertheless we have obtained different results since we have modified the regular expressions and the ontologies entries. However this is not a real difficult for our approach because the characterization process mission is to manage with the incertitude finding the most important hints using the current method and knowledge to do that. Also the results of this characterization process must have a supplementary process, for example a meta-analysis process where an analysis of analysis is performed (Leandro, 2005). Thus, the final impact of these variations is minimal.

5.3.1.7 Discussion: Conclusions and future work of this approach

The present MAS approach has proved to be a good alternative to manage with dynamic changes of information acquisition and upgrades in the knowledge systems. It allows us to create an auto-adaptable modular system which can be modified or upgraded without altering the entire system. We have the prospect of improve this work using dynamic web-services composition similar as is proposed in (Abrougui et al., 2009). This will permit to upgrade dynamically the system at agent level adding new agents to work with different methods or services than the text pattern recognition. As final remark in this appendix approach we have showed a MAS-based approach to characterize an application description by means of a text pattern recognition process and an XML-based standard called ADMACA. The emergent goal of this approach is to be useful to the engineers as a process that assists a MAS-based software engineering process.

5.3.2 Meta-model Features Manual Characterization

The second input required by the matching engine are the building blocks that are essential to build a solution. In our particular problem we have several MAS-based meta-models. Such meta-models act like pieces of solution, nevertheless, we can not employ them directly as meta-models within the matching engine. We also need to characterize them in order to make them machine readable. Such characterization process is less difficult than the AS-characterization process because in this case we have existing definitions created in the overall methodologies works. Therefore we must abstract them from the methodologies.

5.3.2.1 Manual Meta-model acquisition method definition

In our case we have defined textual features of several meta-models in order to consider such textual features as meta-model descriptions. Each feature is related within a domain using a value. Moreover the overall values, domains and feautres are stored within a knowledge base.

The meta-models candidates set is selected using the micro-array data, gathered throughout the AS-characterization process, as a filter. Therefore the resulting candidates subset is composed by several meta-model characterization micro-arrays. In this way the meta-models micro-array characterized representation becomes machine readable. It makes easy to match a AS-characterization micro-array with several MAS Meta-model micro-arrays, it also makes easy to evaluate the compatibility between Meta-model micro-arrays in order to combine them into meta-model groups that could work together.

For this thesis we have defined a manual way to characterize the MAS meta-models into micro-arrays (See the section 7.2.1.2). First, we have chosen a small set of MAS meta-models from existing solutions, for our test case from GeDA-3D (Ramos et al., 2005) and an auction framework (Milidiu et al., 2003), considering the vowels approach. Then for each meta-model we have identified a small set of features that best describe each feature. We have defined 3 domains considering the focus of our test cases. Then we have defined each feature employing a text sentence for each. Finally we have made a survey like an experience acquisition process to gather some statistical information about the performance of each meta-model within each domain.

5.3.2.2 Motivations and satellite phase future

We have chosen to do that in such minimal way because the matching engine requires a characterized input about the building blocks nevertheless this is not the thesis main goal. Then in such a manner we can gather a minimal amount of experience data directly from the experienced humans (from the MAS group team members). This allows to create a starting experience from which the matching engine can make decisions about reliability according to the problem to solve and compatibility between meta-models.

Nevertheless we know that there is a long way to follow to improve such area. We know that one possibility is to create an automated meta-model and features recognition from the existing methodologies. There is also the option to build an automated approach to experience data acquisition from existing MAS solutions. However this thesis do not seeks to follow such topics, moreover, we propose a matching engine that employs inputs from this. That is why we have to define a minimal way to do that.

5.4 Linking Meta-analysis with MAS Software Engineering

Arriving at this point the question is how to link the meta-analysis to the MAS software engineering process to improve the decision-making into the software engineering methodologies for MAS?

To give a response to this question we explain through the following lines, as clearly as possible, how we made the analogy of the classic medical meta-analysis to a meta-analysis that assists in decision-making within the context of software engineering for MAS.

5.4.1 Meta-models and Treatments

The treatment of meta-models in our approach is similar to the evaluations of the substances or medicaments and their efficacy, when they are present in a valid MAS-based solution. As it occurs in a medical meta-analysis when a treatment shows its efficacy for an illness. It means that we can evaluate two different meta-models of the same kind (for example Agent) to check which one is the best for the requested solution, it is analogous to the valuation of two different medicaments against the same illness.

5.4.2 Domain and Illness

We propose to identify the domain of solution. In other words, each meta-model has a value of efficacy within a domain of solution, therefore, it means that a meta-model could be efficient within some domain of solution, as well, could be inefficient in a different domain of solution. Such domain is comparable with a specific illness where a medicament could be efficient or not. On the other hand to identify the domain of solution from a meta-model we propose to characterize each meta-model using a set of abstracted characteristics that describes the meta-model and links each characteristic to a domain with an efficacy value. After that, store all these data into a knowledge base.

5.4.3 Application Specification and Disease Description

For our meta-analysis the application specifications (AS) is like a disease description. Thus, we propose to characterize the AS identifying the problem characteristics and domain. Such characterization can be performed, for example, through the text pattern recognition. So, it acts like the data gathering protocol. Therefore, such characterization process permits to know the AS problem characteristics and consequently their domain of solution. Such data is something like the illness identification and its information guide us to find which the most adequate meta-models are. By the way, locating the domain of the problem acts as a discrimination factor reducing the number of possible domains of solution. It permits to select the meta-models that have high efficacy values in the selected domain, therefore, the most promising ones.

5.4.4 AS-Characterization and single trial

Therefore the equivalent of a single trial is: the AS characterization, the characterization protocol, the AEIO meta-models - that match with the AS - and the efficacy values - that link the meta-models characteristics with the AS characterization. However an individual or single trial in the

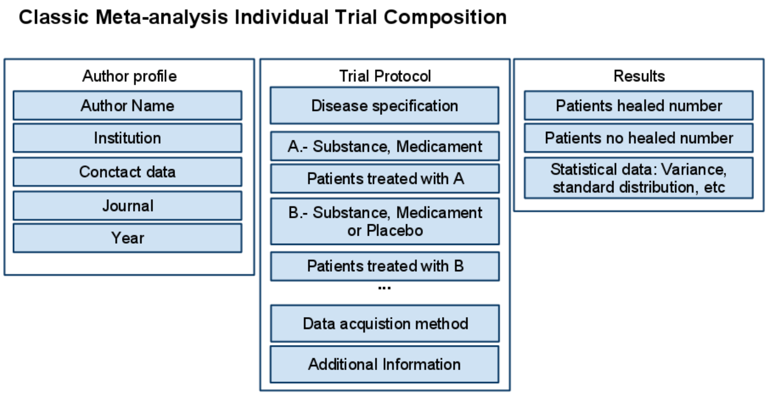


Figure 5.18: Classic Meta-analysis trial composition

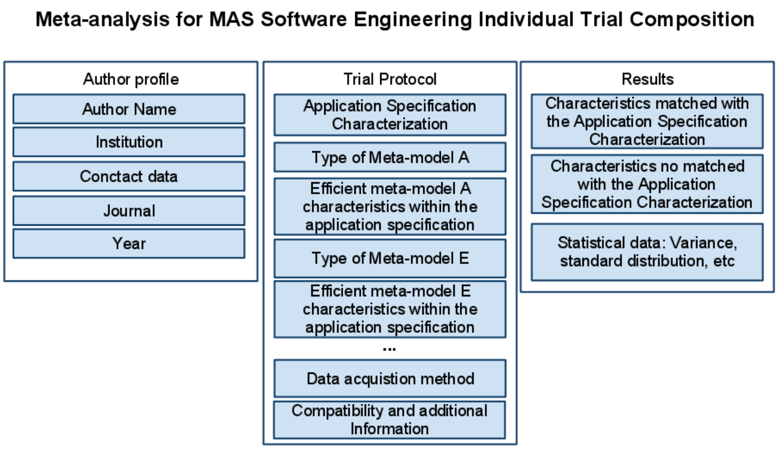


Figure 5.19: Meta-analysis for MAS software engineering trial proposed composition

medical meta-analysis context is similar as the showed in the fig. 5.18, consequently we propose the meta-analysis for MAS software engineering trial in the fig. 5.19.

5.4.5 Meta-analysis for MAS Software Engineering

The meta-analysis process applied to the MAS software engineering seeks to assist the automated making decisions providing the statistical functions and data required. Such automated making decisions issue is closely tied with the Artificial Intelligence (AI) field. Thus the solution architecture, as follows in the next section, is built over the meta-analysis, AI and software engineering. Those areas play key positions in the solution architecture. Nevertheless in this article we focus particularly in the AI and how it works using meta-analysis.

Chapter 6

Matching Engine

6.1 MAS Overview

The proposed approach is MAS-based therefore we can describe it using the vowels approach:

- Agent: The agent's kind is cognitive because they make decisions individually and representing a MM-characterization.
- Environment: Semi-observable at MAS level because we can observe the groups organized by each agent.
- Interaction: Is based on negotiation through messages like dialogs to reach groups memberships and to choose winning groups.
- Organization: The organization is based on the making decision process, we can locate it as a self-organization based on patterns because there are rules to follow but they depend on the chosen meta-models (represented by agents) in order to follow these rules and build the organization (As we described in section 1.4.3.1).

The comparison engine is built using a MAS-based solution to perform the data meta-analysis and matching MMs and infer solutions sets. The agents employed are cognitive; they have the skill of making decisions individually. Moreover each agent acts representing a MM-characterization. The environment is semi-observable (see fig. 6.1) thus it enables to observe the group formation through the MAS performance. The interaction is based on message passing between agents similar as the contract net protocol (FIPA, 2002) but searching for suitable group memberships. Finally the organization is based on dynamic group creation, where each group of agents represents a proposed solution. The fig. 6.1 part C shows the MAS architecture overall.

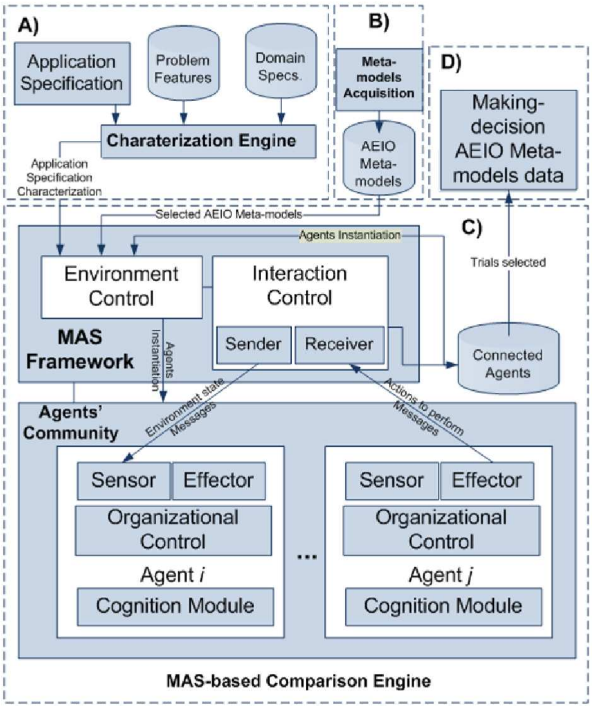


Figure 6.1: MAS Architecture Overview

6.2 Proposed Solution Architecture

To achieve the automated making decisions we propose an architecture composed by an AS characterization engine - that translates the AS into an abstract problem characteristics and chooses a domain (see fig. 6.1 part A) - and a comparison engine - that matches the AS characterization data (see fig. 6.1 part C) with the meta-models (MM) characterization (see fig. 6.1 part B) and propose sets of possible meta-model-based solutions (see fig. 6.1 part D) -. However in this thesis we focus mainly in the matching process performed in the comparison engine, nevertheless, we address the AS and MM characterization process to clarify the complete trial process creation. Analogically we propose the meta-analysis for MAS software engineering process - showed in the fig. 6.2 at the highlighted box - using probabilistic Artificial Intelligence as the present thesis main contribution.

6.3 Comparison engine

Through the proposed solution process, the first step - the AS-characterization process (see fig. 6.1 part A) - provides the AS-micro-array as result. Such micro-array information is required in the second stage - in the comparison engine (see fig. 6.1 part C) - in combination with the MM-characterization KB data. Both information sources feed the matching process inside the comparison engine. We have an important remark to justify the use of a comparison engine in this approach: At present we have no database of existing studies with trials about the efficacy using MAS components with quantitative results as occurs in a meta-analysis. In other hand we have a lot of MAS methodologies composed by models, components, development process, etc. (Jorge Gómez-Sanz, 2004) that have proved to be useful in certain domains. So, we consider that we can extract from the existing solutions their representing meta-models, similar as is proposed in (Z.Guessoum and Jarraya, 2005), and evaluate the efficacy of each one of them in each solution domain. Also is possible to automate the refinement of the efficacy values accuracy and make them more precise each time considering the user feedback as part of a automated learning process. In this way we can create an equivalence process of what is Pubmed for the medical related meta-analysis but in the context of MAS software engineering. So, that's why we propose to create a comparison engine capable to combine meta-models using the experience lodged in his MM-characterizations KB to infer which of these meta-models could work together. Therefore the matching process is a benchmarking method that compares both characterization sources - AS-characterization and KB of MM-characterizations - with the aim at selecting the most adequate meta-models and proposing a combination of them as a candidate solution for the application specification. Thus, the matching process is conceived within a comparison engine. Such a comparison engine is built using a MAS approach that we will describe briefly in the section 6.4. However, in this article we focus in the probabilistic artificial intelligence employed behind the automated making decisions described in the section 6.5.

6.4 MAS Architecture

The comparison engine is built using a MAS-based solution to perform the data meta-analysis and matching MMs and infer solutions sets. The agents employed are cognitive; they have the skill of making decisions individually. Moreover each agent acts representing a MM-characterization. The environment is semi-observable (see fig. 6.1) thus it enables to observe the group formation through the MAS performance. The interaction is based on message passing between agents similar as the contract net protocol (FIPA, 2002) but searching for suitable group memberships. Finally the organization is based on dynamic group creation, where each group of agents represents a proposed solution. The fig. 6.1 part C shows the MAS architecture overall.

6.4.1 Agent Architecture

The agent architecture comprises the next components (see also fig. 6.3):

- Self Representation. It stores the agent self-representation composed by the AS-characterization and the represented MM-characterization.
- External Representation. It comprises the third agents representation; it is the known agents MM-characterization data. They are stored in this module; a known agent is an agent that has contacted the host agent, regardless of whether that agent has been included to the host agent group or not.

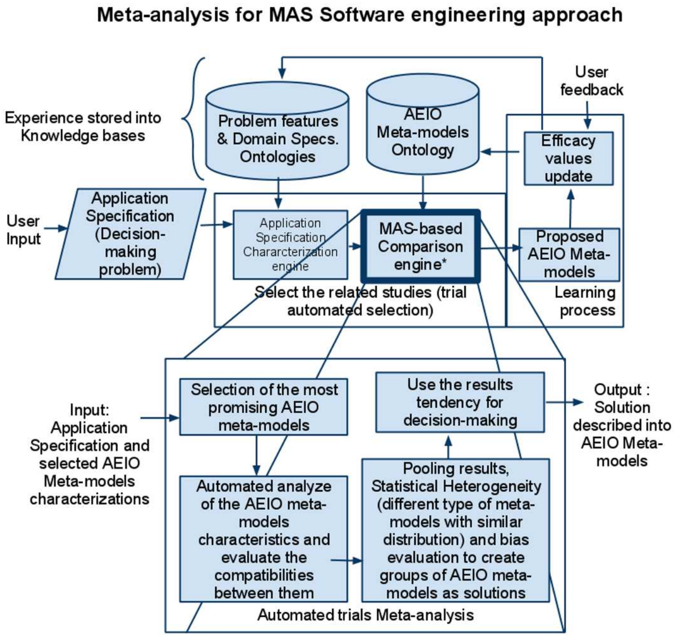


Figure 6.2: Meta-analysis for MAS software engineering process overview

- Organizational control. This control belongs to the organizational component of the entire MAS; it allows to the environment observe the groups created by means of agent interactions. It comprises the agent group registry sub-module that stores the group membership information of each agent. As we explain further in the section 6.4.2 each agent performs a negotiation with the rest of the agents to evaluate the group membership, therefore, this module stores the structures of the groups created by each agent as results of these agents interactions.

- Agent interaction control. This control allows the agent to interact with other agents communicating them. It is composed by a messaging module that acts as sender and receiver. The protocol employed to communicate is similar to the contract net interaction protocol because the agent negotiates the group membership with other agents.

- Cognition Control. The main task of this control is to make decisions. It comprises the Induction-based making-decisions module together with the negotiation module. The negotiation module creates and interprets the required messages to negotiate the group memberships. On the other hand the induction-based module owns the induction algorithm that evaluates the next actions (See also the fig. 6.4):

  - Assess the host agent MM-characterization compatibility with the AS-characterization.
  - Evaluate an external agent MM-characterization to reject it or accept it into the local group considering the compatibility with the host agent and the current members of the

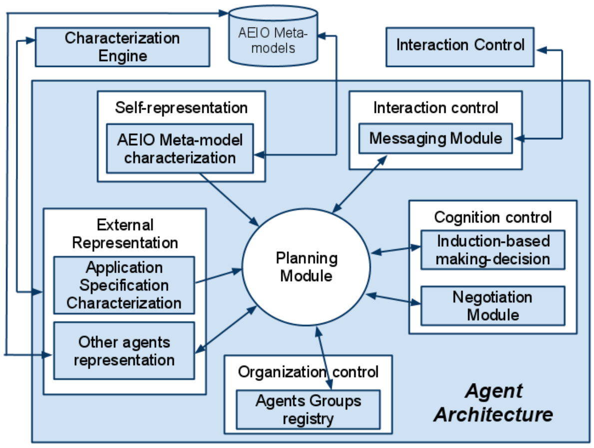


Figure 6.3: Agent Architecture Overview

group.

In the section 6.5 we deeply explain how the probabilistic AI works to make such decisions.

- Planning Module. Such module performs the host agent behavior using all the agent resources that have been described. Consequently this module controls the agent actions to perform in the time line. It keeps the control of the agent using all the modules and controls to create the agent's behavior.

6.4.2 MAS Operation

In the MAS working process each agent represents a MM-characterization from a subset selected from the KB of MM-characterization. The agents perform together the MM-groups formation task. The agents pass messages between them; thus, compare their MM-characterization and individually making decisions about the memberships. The groups emerge from the point of view of each agent self-representation. Each agent compares their MM-characterization micro-array -self representation - with others of different type to select the most appropriates. Such groups are created using the vowels approach (Demazeau, 2001), therefore, each agent searches for a type other than their own. At the end each agent has created a group; therefore, the one that is most likely to be a solution is chosen. The criterion is: First rejecting the groups that do not contain the type agent. Moreover, for each group, estimate the probability of the entire group evaluating the probability of all the members together. Finally choose the group with the most likely probability value. This is a straightforward way to apply the meta-analysis of characterized data. Nevertheless and as we will discuss later in the conclusions section is possible to store the successful groups results into a knowledge data base as historic or experience data.

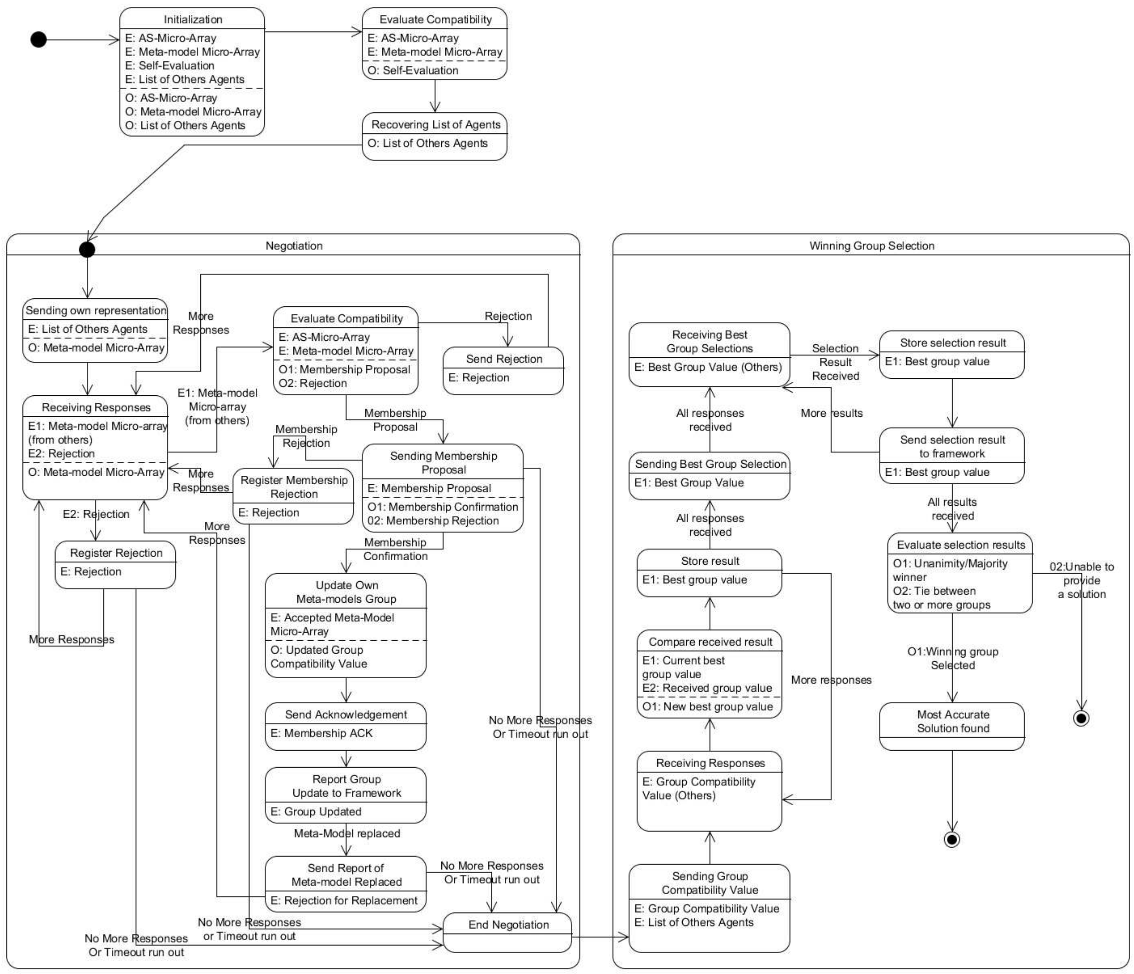


Figure 6.4: Agent Behavior State Transition Diagram

6.4.2.1 Agent Behavior

The individual agent's behavior has three main branches (See fig. 6.4) :

- Initialization. This is the first step in the agent instantiation. It receives the AS and MM micro-array data and makes a compatibility self evaluation. Such an initial phase also recovers the existing agent list in order to contact them further.

- Negotiation. This is the second phase in the agent behavior. Such a step performs the negotiation with the others agents in order to build a solution group in conformity to the owned meta-model compatibility. The negotiation comprises:

  – Send own representation. It sends the owned Meta-model micro-array to every compatible[1] agent in the MAS.

[1] A compatible agent is an agent of different vowel Meta-model type.

- Receiving responses. After sending the own representation the agent pushes up the receiving responses thread in order to catch all the micro-array representations sent by other agents.
- Evaluate Compatibility. When a micro-array representation is received the first action is evaluate the compatibility with the own representation.
- Send Rejection. If it is not compatible then a rejection response is triggered.
- Send Membership proposal. If it is compatible then a membership proposal is sent to the evaluated agent.
- Register rejection. When a rejection is received (as response to the own representation) then the meta-model that originated such a rejection is registered.
- Register Membership rejection. This state is achieved when a membership rejection is received (as response to a Membership proposal) then the meta-model that originated it is registered.
- Update Own Meta-models group. This state arrives when a membership confirmation is received (as response to a Membership proposal) then the meta-model group owned is updated adding the new compatible meta-model member.
- Send Acknowledgment. When the meta-model group has joined the new member a acknowledgment message is sent to the new meta-model member agent.
- Send Notification of replaced meta-model. This state is reached if the meta-model added to group has replaced an existing (same type) meta-model. If so, then a rejection message is sent to the old meta-model group member notifying the change.
- Report Group Update to Framework. This state is achieved once the membership has been processed and it notifies to the framework the current state of the owned group. It works especially to made this information available to external observations.
- End Negotiation. This is the negotiation final state achieved when there are not more messages to process or when the timeout has run out.

• Winner Group Selection. This is the third and last phase where the group results of the previous sections are shared and thus a winner is voted to being selected.

- Sending own Group Compatibility Value. This is the initial state of the group selection where the resulting group compatibility value[2].
- Receiving Responses. After sending the own group compatibility value the agent rises a state that keeps awaiting for a message reception.

[2]The resulting group is evaluated collectively with the AS-micro-array in order to find the compatibility value with the entire group

- Compare received result. When a message is received this state is activated in order to compare the received compatibility result with the current best one found. If it is initial then it is compared with the own group compatibility result.
- Store result. This state is reached when the received result is compared and then such a result is stored.
- Sending Best Group Selection. When all the group compatibility values has been received then this states is achieved. This state sends a vote of winning selection group.
- Receiving Best Group Selections. This state follows the sending group selection in order the receive the selection votes of the other agents.
- Store selection result. This state stores the selection votes results received.
- Send selection result to framework. It shares the results received to the framework in order to make it observable.
- Evaluate selection results. It count all the results voted when all the votes has been received and then publish the results. If there is a tie between two or more groups then is unable to provide a solution.
- Most Accurate Solution Found. If there is a winner by majority then this final state is achieved. The winner group is published by all the agents and by the framework.

The interaction diagram 6.5 shows the message passing between agents when they are collaborating together and forming groups. The last step starts when each agent has already evaluated all the membership options to create a group (or when the time has run out) finally the group 6.6

6.4.3 Knowledge bases

There are fifth different knowledge bases; three of them are explicit and already stated, they are the next: AS-characteristics, domains abstractions and MM-characterizations. There are two more that are implicit and stores memory or experience values: The MM-instantiation memory and the MM-unification memory. We explain how they are composed in the section 6.5.

6.5 Probabilistic AI for making-decisions

When the comparison engine MAS is running the agents create groups interacting between them and thus performing the meta-analysis of data to make decisions about the group membership. Consequently each agent has a group that is being evolving through the execution. We can observe such group evolution in a certain point of the execution time line; we consider each one of the groups created as possible solution from the point of view of each agent's represented meta-model. Considering that each member of the group is a meta-model that has been characterized the making decision process within each agent uses the efficacy information values of each MM-characterization

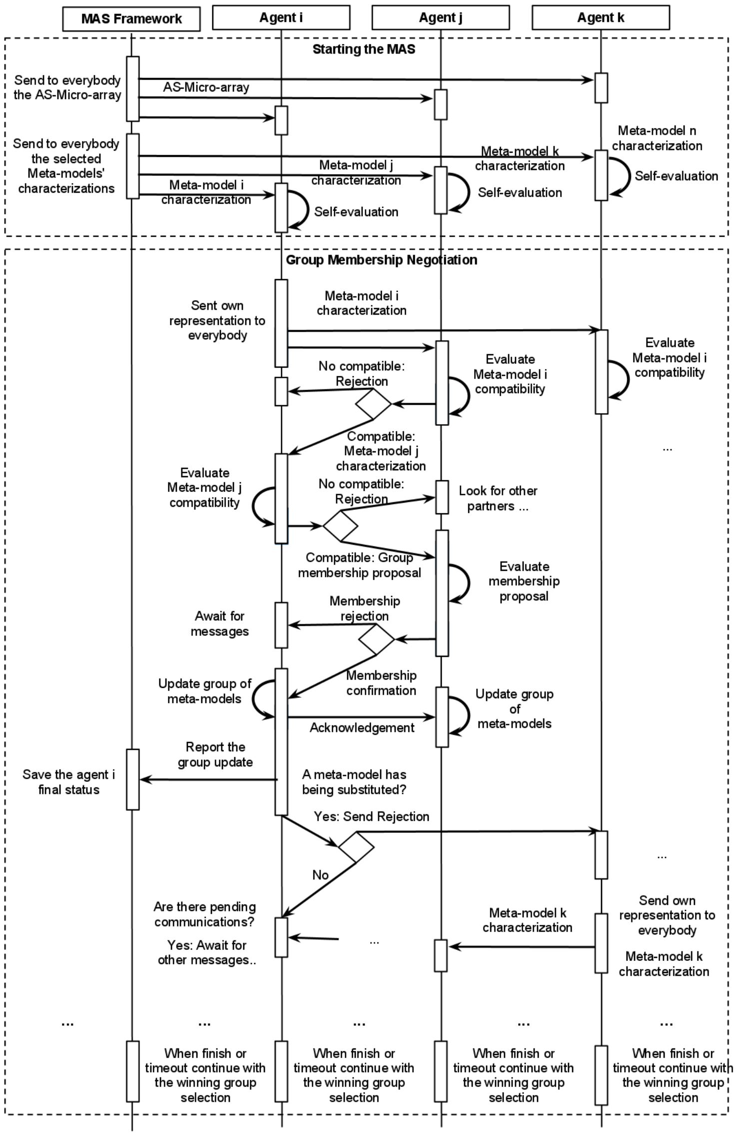


Figure 6.5: Agent Initial Phase and Group Negotiation Collaboration Diagram

to perform the meta-analysis. This step substitutes the lack of full solution trials that are employed in a classical meta-analysis because each agent creates a candidate solution trial when they build its group. The agents final step is to decide the best group among all.

6.5.1 Experience KB

The MM-instantiation memory stores values about the matching results of AS-characterization and MM-characterizations micro-arrays. It means that it stores - for each AS-characteristic and each MM-feature and under each domain - the following data:

- Total times when the matching was enable; as well the total times when was disable. For each

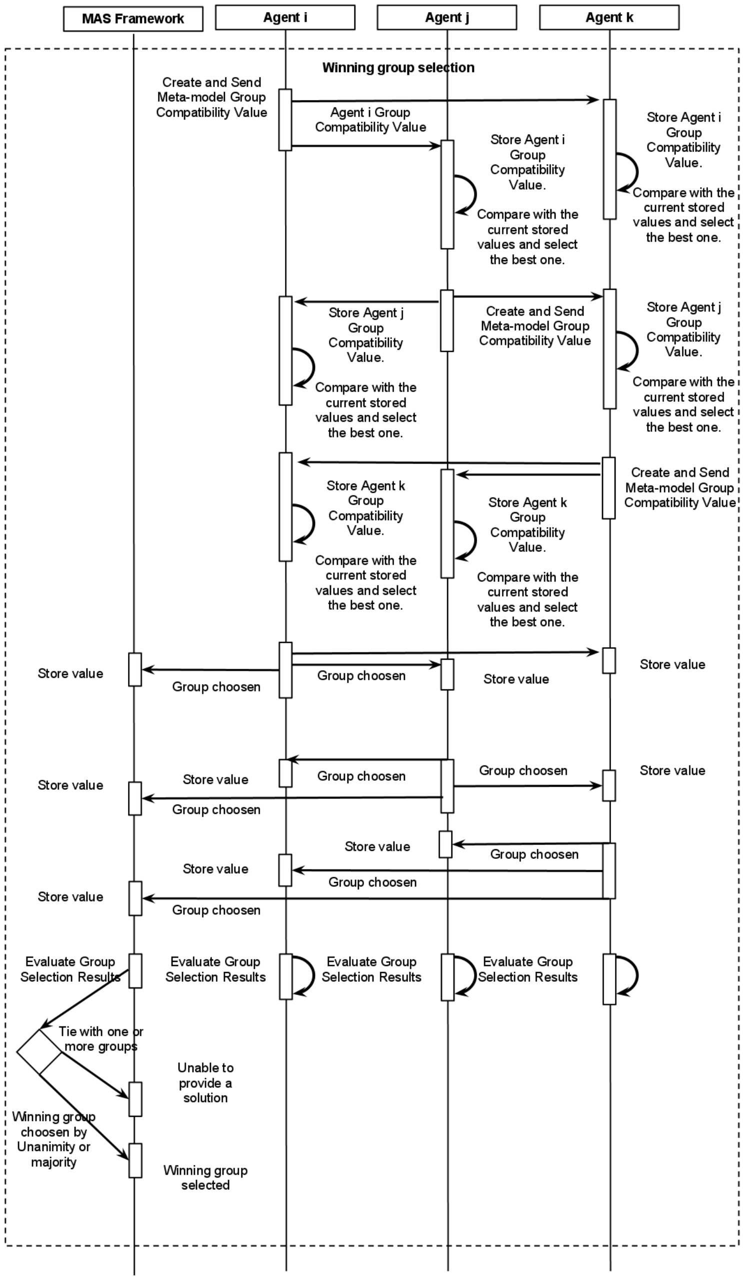


Figure 6.6: Agent Final Step Winning Group Collaboration Diagram

matching possible value and considering one selected domain:

– Total times when some AS-characteristic was not efficient and some MM-feature too in the domain.

– Total times when some AS-characteristic was not efficient and some MM-feature was efficient in the domain.

– Total times when some AS-characteristic was efficient and some MM-feature was not efficient in the domain.

- Total times when some AS-characteristic was efficient and some MM-feature too in the domain.

Moreover the MM-unification memory stores values about the solution results of MM-characterization micro-arrays. In other words it stores - for each pair of different MM-features under each domain - the following data:

- Total times when both MM-features were part of a solution and as well total times when were not. Therefore for each situation count:
  - Total times when both MM-features were not efficient in the domain.
  - Total times when one of the MM-features was efficient and the other not in the domain.
  - Total times when one of the MM-feature was not efficient and the other was efficient in the domain.
  - Total times when both MM-features were efficient in the domain.

Both information KBs are stored as two different hyper-matrices [3]. The unification rules are related to the construction constraints between the different MM-features. Such rules are defined or updated for each MM-feature at the moment of adding a new meta-model. Furthermore these benchmarking values could be acquired or updated through learning processes and depending on the solutions provided by the approach proposed. It means that at each time that the approach is employed to provide a solution the making-decision has more "experience". Consequently, the approach will provide most suitable solutions. However in our example we have employed hand-written values according to the laboratory member's experience. Both experience KB are improved in the identification process that are part of the Bayesian program employed and explained in more detail in the next section.

6.5.2 Bayesian program definition

As explained in (Bessière et al., 2008) commonly a Bayesian program is defined using the next structure:

- Description. It is a probabilistic model about some phenomenon that is obtained from the next two branches:
  - Specification. Such specification expresses the modeled phenomenon knowledge in the following probabilistic terms:
    * Variables. All the important and known variables related to the phenomenon.

[3] A matrix of many matrices levels

    * Decomposition. Is the joint distribution of the variables. Usually is done using a decomposition that keeps the joint distribution as a product composition of simplified distributions.
    * Forms. To compute the joint distribution we must specify all the distributions appearing in the decomposition with all the possible values for each variable.
  - Identification. Is the learning phase of the probabilistic where the initial data is refined and becomes more accurate.

- Questions. The questions are defined by branching a set of variables in three subsets: the searched variables (on the left side of the conditioning bar), the known variables (on the right side of the conditioning bar) and the free variables. Such questions must be answered using the decomposition and forms definitions.

6.5.3 Choosing MM-Characterizations

To choose the most promising the MM-Characterizations for the AS we propose using a Bayesian-based inference algorithm because it allows to evaluate the probability of solution using certainty values. The certainty values are similar to the efficacy values employed in the MM-Characterizations and AS-Characterization micro-array. Thus is possible to evaluate any MM-Characterization as partial solution using itself and the AS-Characterization data. This process allows choosing a subset of MM-Characterizations as candidates to create representative agents into the MAS that executes into the matching engine. Moreover to achieve such selection we have defined a Bayesian program as follows:

6.5.3.1 Specification

Our phenomenon is about to match an AS-characterization with a MM-Characterization. Considering the explained context in the last sections the variables identified are the next:

- $Match$. A boolean variable that specifies if some AS_k and some MM_{τ_k} match or not.
- Considering only one domain δj each time we define both characterizations:
  $AS'_i = \left\{\theta_{j,1}, \ldots, \theta_{j,c'}\right\}$ for simplicity $AS'_i = \{\theta_1, \ldots, \theta_{c'}\}$.
  $MM'_{\tau_k} = \left\{\mu_{j,1}, \ldots, \mu_{j,f'}\right\}$ for simplicity $MM'_k = \left\{\mu_1, \ldots, \mu_{f'}\right\}$.
- Therefore we define the set of matching pairs as $L_{i,k} = AS'_i \times MM'_k$, moreover we have the AS-characteristics and MM-features values:
  $L_{i,k} = \left[(\theta_1, \mu_1), \ldots, (\theta_1, \mu_{f'}), \ldots, (\theta_{c'}, \mu_1), \ldots, (\theta_{c'}, \mu_{f'})\right]$.
- Considering that each $\lambda_j = (\theta_i, \mu_k)$ for simplicity:
  $L_{i,k} = [\lambda_1, \ldots, \lambda_l]$ where the size of the set is $l = c' \times f'$

We define the decomposition in the next manner:

- First we define the top level joint distribution: $P(Match \wedge L_{i,k})$
- Such distribution is equivalent to: $P(Match \wedge \lambda_1 \wedge \ldots \wedge \lambda_l)$
- Decomposing it we obtain:
  $P(\ Match\ ) \times P(\ \lambda_1 \mid Match) \times P(\ \lambda_2 \mid Match \wedge \lambda_1)$
  $\times \ldots \times P(\ \lambda_l \mid Match \wedge \lambda_1 \wedge \ldots \wedge \lambda_{l-1}\ )$
- We assume that each match between each pair composed by λ_i is independent of the rest, therefore, we can simplify the joint distribution:

$$P(\ Match \wedge \lambda_1 \wedge \ldots \wedge \lambda_l) = P(\ Match\ ) \times \prod_{i=1}^{l} P(\ \lambda_i \mid Match)$$

However we must state that each λ_j is related to a pair (θ_i, μ_k).

Consequently, to calculate the probability values of the joint distribution we must define the parametric forms using all the possible values of each variable. First, we exchange all the real values of the characterization variables (θ_i, μ_k), contained in each λ_j, using a statistical model. Such model is defined the values from 0 to 0.7 as $false$ and $true$ the rest, 0.7 to 1. Therefore such values are employed to define the following forms:

First we define for the variable $Match$:
$P(\ Match\ ):\ \ P(\ [Match = false]\ ) = 0,3\ \ P(\ [Match = true]\ ) = 0,7$
Therefore for each of the l forms of $P(\ \lambda_j \mid Match\ )$ and considering that $\lambda_j = (\theta_i, \mu_k)$ we have $P(\ (\theta_i, \mu_k) \mid Match\ )$.

$[\lambda_j]$	$[Match]$	value
$false$	$false$	$1 - \frac{1+n_f^j}{2+n_f}$
$false$	$true$	$1 - \frac{1+n_t^j}{2+n_t}$
$true$	$false$	$\frac{1+n_f^j}{2+n_f}$
$true$	$true$	$\frac{1+n_t^j}{2+n_t}$

Where each n_{Match}^j counts the number of true or false matches for the pair λ_j, such pair has a AS-characteristic θ_i and a MM-feature μ_k. Moreover each n_{Match} value counts the total times of true or false matches cases has been found.

The last parametric forms are based in the succession rule defined by Pierre-Simon Laplace (Laplace, 1812). Such rule allows us to infer the probability of finding again a proposed combination of values using only a few values. Normally the meta-analysis needs a high amount of data to work, however, we have not all the enough data at the moment. That is why we use such rule instead of a form based into a more related meta-analysis method.

In order to reach accurate results at this stage the proposed identification process has been developed using a survey that has been answered by some MAS experienced members of our laboratories.

The corresponding values of the experience KB (explained in the section 6.5.1) are updated using the user feedback.

6.5.3.2 Questions

This Bayesian program seeks to infer if some AS-characterization can be matched with some MM-characterization. Therefore the related questions are:

$$P(\, Match \mid \lambda_1 \wedge \ldots \wedge \lambda_l)$$

that is equivalent to:

$$P\left(\, Match \mid \; (\theta_1, \mu_1) \wedge \ldots \wedge \left(\theta_1, \mu_{f'}\right) \wedge \ldots \wedge (\theta_{c'}, \mu_1) \wedge \ldots \wedge \left(\theta_{c'}, \mu_{f'}\right)\right)$$

also such question could be solved in the next manner:

$$P(\, Match \mid \lambda_1 \wedge \ldots \wedge \lambda_l) = \frac{P(\, Match \,) \times \prod_{i=1}^{l} P(\, \lambda_i \mid Match \,)}{\sum_{Match} P(\, Match \,) \times \prod_{i=1}^{l} P(\, \lambda_i \mid Match \,)}$$

6.5.4 Building Solution Groups: Evaluating MM-Characterizations

Then to know the successful probability of using two different meta-models M_i and M_j as part of the solution we use the following formula (representation and group membership evaluations, both of external agents) to evaluate the meta-models compatibility and consequently decide to add or not to the solutions group:

6.5.4.1 Specification

In this case our phenomenon is the MM-characterization group conforming as partial solution for AS-characterization. Therefore we have identified the following variables:

- Solution
- A selected domain δ_j. Same as 6.5.3.1.
- Two sets of MM-characterization from different MMs types:
  $MM'_{\tau_i} = \left\{\mu^{\tau_i}_{j,1}, \ldots, \mu^{\tau_i}_{j,f'}\right\}$ for simplicity $MM'_i = \left\{\mu^i_1, \ldots, \mu^i_{f'}\right\}$.
  $MM'_{\tau_l} = \left\{\mu^{\tau_l}_{j,1}, \ldots, \mu^{\tau_l}_{j,f'}\right\}$ for simplicity $MM'_l = \left\{\mu^l_1, \ldots, \mu^l_{f'}\right\}$.
- Thus we define the set of solution pairs as $S_{i,l} = MM'_i \times MM'_l$, consequently we have both set of MM-features values:
  $S_{i,l} = \left[(\mu^i_1, \mu^l_1), \ldots, \left(\mu^i_1, \mu^l_{f'}\right), \ldots, \left(\mu^i_{f'}, \mu^l_1\right), \ldots, \left(\mu^i_{c'}, \mu^l_{f'}\right)\right]$.

- Considering that each $\sigma_j = (\mu_p, \mu_k)$ and $\mu_p \neq \mu_k$ then:
  $S_{i,l} = [\sigma_1, \ldots, \sigma_s]$ where the size of the set is $s = f'^i \times f'^l$

Therefore the decomposition is defined as follows:

- First we define the top level joint distribution: $P(Solution \wedge S_{i,l})$
- Such distribution is equivalent to: $P(Solution \wedge \sigma_1 \wedge \ldots \wedge \sigma_s)$
- Decomposing it we obtain:
  $P(\ Solution\ ) \times P(\ \sigma_1 \mid Solution) \times P(\ \sigma_2 \mid Solution \wedge \sigma_1)$
  $\times \ldots \times P(\ \sigma_s \mid Solution \wedge \sigma_1 \wedge \ldots \wedge \sigma_{s-1}\ )$
- Assuming that each match between each pair composed by σ_j is independent of the rest, therefore, we can simplify the joint distribution:

$$P(\ Solution \wedge \sigma_1 \wedge \ldots \wedge \sigma_s) = P(\ Solution\ ) \times \prod_{i=1}^{s} P(\ \sigma_i \mid Solution)$$

To define the parametric forms we consider a statistical model where the values from 0 to 0.7 as $false$ and $true$ the rest, 0.7 to 1. Thus we define for the variable $Solution$:
$P(\ Solution\ ):\ \ P(\ [Solution = false]\ ) = 0,3\ \ P(\ [Solution = true]\ ) = 0,7$
Also we define the same for each of the s forms of $P(\ \sigma_j \mid Solution\ )$ and considering that $\sigma_j = (\mu_p, \mu_k)$ we have $P(\ (\mu_p, \mu_k) \mid Solution\ )$.

$[\sigma_j]$	$[Solution]$	value
$false$	$false$	$1 - \frac{1+n_f^j}{2+n_f}$
$false$	$true$	$1 - \frac{1+n_t^j}{2+n_t}$
$true$	$false$	$\frac{1+n_f^j}{2+n_f}$
$true$	$true$	$\frac{1+n_t^j}{2+n_t}$

Each $n^j_{Solution}$ counts the number of true or false solutions found for the pair σ_j that comprises the MM-features μ_p and μ_k. And each $n_{Solution}$ value counts the total true or false solutions found. Similar as is described in section 6.5.3.1.

6.5.4.2 Identification

In this case the identification process has been developed using a survey just after the matching engine selects the candidates. Such survey has been answered by our laboratories members too. So, the experience values has been collected from the survey gathered data. Thus the corresponding values of the unification memory KB (explained in the section 6.5.1) are updated using the user feedback.

6.5.4.3 Questions

This Bayesian program aims at inferring if two different types meta-models represented by two MM-characterizations could be considered to be joint into a solution . Therefore the related questions are:

$$P(\ Solution \mid \sigma_1 \wedge \ldots \wedge \sigma_l)$$

also such question could be solved in the next manner:

$$P(\ Solution \mid \sigma_1 \wedge \ldots \wedge \sigma_s) = \frac{P(\ Solution\ ) \times \prod_{i=1}^{s} P(\ \sigma_i \mid Solution\ )}{\sum_{Solution} P(\ Solution\ ) \times \prod_{i=1}^{s} P(\ \sigma_i \mid Solution\ )}$$

6.6 Discussion: Present approach contributions

The present work proposes to employ many existing technologies to create a new approach that rises as a preliminary phase before the analysis and design phases for every MAS-based project. Similarly as PASSI (Cossentino, 2005) our approach analysis the problem requirements in order to identify the domain. Nevertheless we identify the problems characteristics from a text description in order to find the domain. Therefore we employ such domain and problem characteristics to map them with the knowledge base of meta-models features to identify meta-model candidates. This is done in a contrary way in relation to the MASSIVE approach (Jürgen, 2001) that proposes to map from the model features to the implementation. We finally create intelligent agents as manager agents that owns a meta-model independently. This is done similar as the software development views proposed in approaches like INGENIAS (Pavón and Gómez-Sanz, 2002) and MASSIVE (Jürgen, 2001) because each agent make decisions according to the view of the type of the owned meta-model. In this case such views are based on the evaluation of the features characterization values of the owned agent with the values of the rest of the compatible agents. The meta-model types have been defined inspired in the approach vowels (Demazeau, 1995).

Part IV

Evaluation

Chapter 7

Tool Implementation

7.1 Tool Specifications

Considering the architecture of our approach we have implemented a prototype using Java, XML and JADE(TILab, 2011) to test it. We have chosen JADE among other options because it is the most well documented and is one of the most mature, active and updated frameworks. Also JADE is a fully functional Java-based extensible framework that give us the enough freedom to adapt it with the text-based recognition and Web services in order to make our tool extensible. Finally JADE is one of the most contributed and active open source MAS frameworks that make it a reliable framework. There are some commercial books about it and several tutorials on the Internet that make it easy to use and quick to learn among others.

So we have created a Graphic User Interface (GUI) where we can load the Knowledge Database of domains, problem characteristics, meta-models features and meta-models. Therefore we can perform or select manually an AS-characterized micro-array. Thus considering the AS-micro-array to select the most promising meta-model candidates. Finally we can launch the MAS where each meta-model candidate is taken by an agent that manages him to create a group of solutions and after of an interaction period to find the most promising group of meta-models as solution. The tool is composed of several modules as seen in the UML diagram fig. 7.1.

The UML diagram displayed in fig. 7.1 represents the different modules of the system. As we can see the Tool module is the start point from which we have two main paths: Create a KB using the KB Creator module or start the tool process using the Matching Engine Tool. KB Creator Module allows us to edit and create data about the problems characteristics, meta-models features and domains and the values that relates each characteristic/feature with a domain. The KB Loader Module allows to load within the Matching Engine Tool such KBs created. The Matching Engine Tool comprises the sub-modules:

- Meta-model Candidate Chooser. It allows to select Meta-model candidates from the KB in order to employ them within the matching Engine.

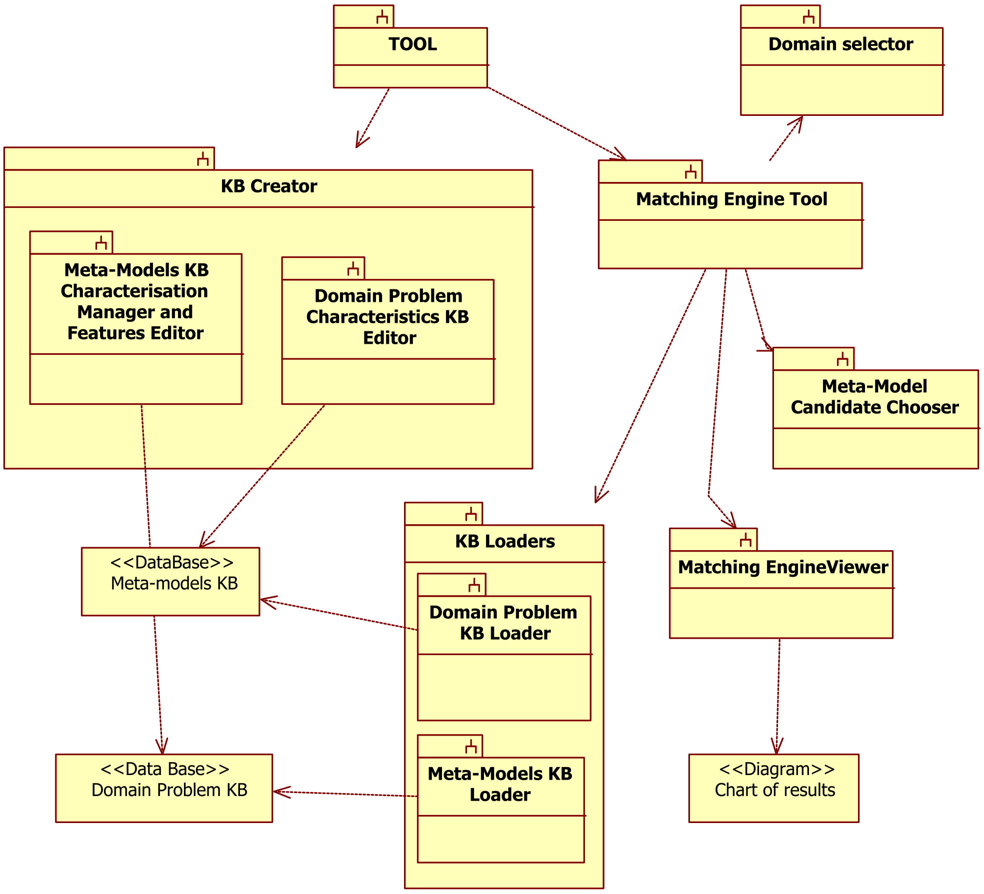


Figure 7.1: Prototype tool UML diagram.

- Matching Engine Viewer It allows to see the Matching Engine performance at each negotiation step and until the end of the process showing the final group selection.

- Domain Selector. It selects the problem domain and generates the AS-micro-array.

7.1.1 Activity Diagrams

We defined the prototype main activity diagram displayed in the fig. 7.2. Such a main diagram follows a path that requires an Application Specification characterization stage that comprises identification of domain and problem characteristics, therefore a selection phase where the AS-micro-array is taken to find a solution domain and to match the problem characteristics with the entire set of meta-model features.

In the activity diagram presented in the fig. 7.3 we represent the behavior path that is performed for each agent within the MAS. The behavior diagram displays step by step how the agent built the best compatible group considering the meta-model that the agent represents.

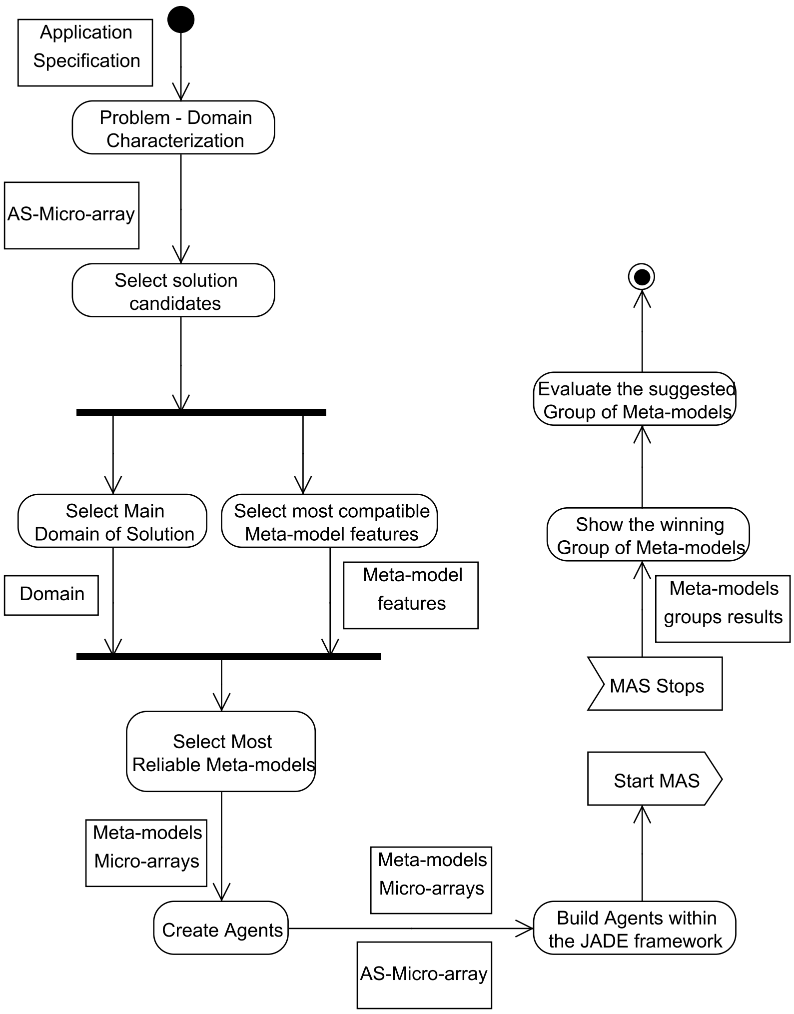


Figure 7.2: Main Activity Diagram

7.1.2 Use cases

The uses cases of the prototype are detailed in the next sections:

7.1.2.1 MAS Meta-analysis

The meta-analysis is performed receiving first the Application Specification data from the user. In this case its a file provided by the AS Editor.

- AS-Characterization. The Application Specification is analyzed and then characterized in order to find the Problem Characteristics and domain. This is done considering the information provided within the file. The result of such an analysis is the AS-Micro-array.
  - Find Characteristics. The Application specification is analyzed considering a Knowledge base of Problem characteristics and the Word Net structures. The result is a list of problem characteristics with a relation value.

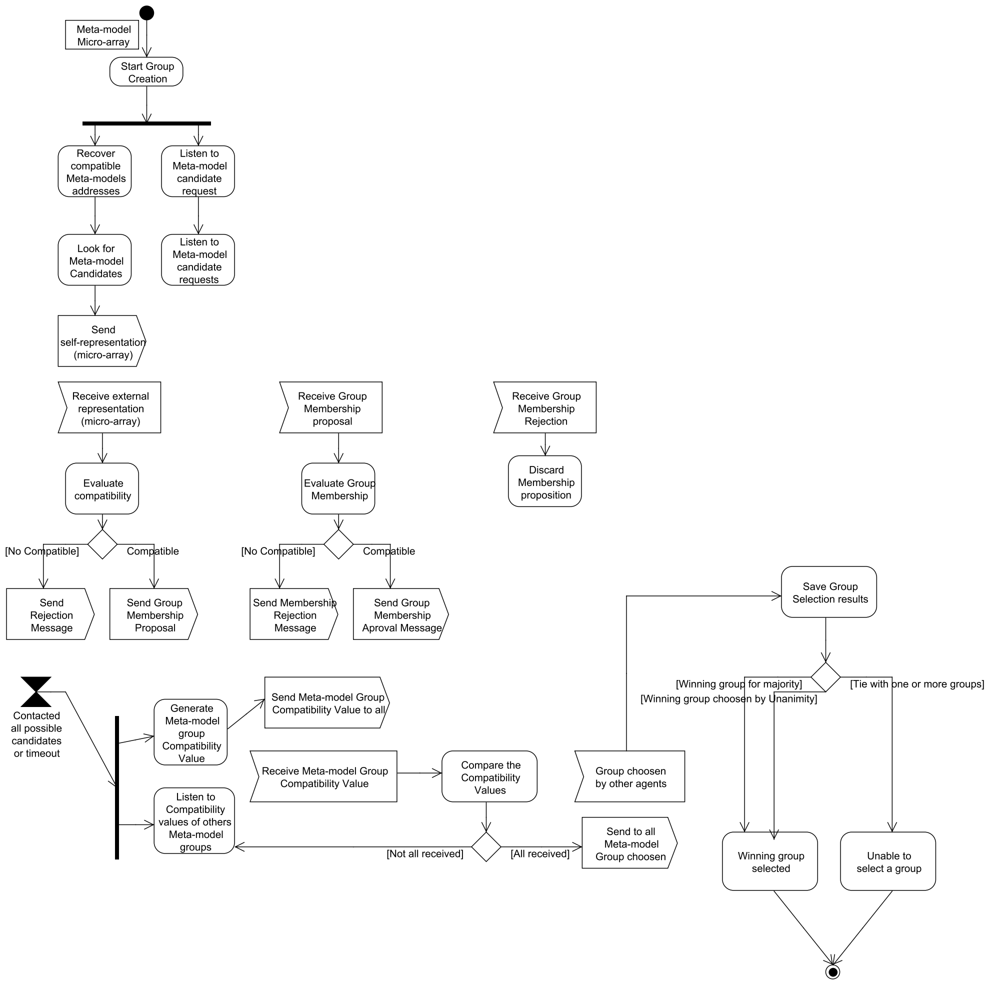


Figure 7.3: Agent Cognitive Behavior Activity Diagram

- Find Domain. Using the list of problem characteristics found the Knowledge base of Problem characteristics is employed in order to calculate statistical values and determine the Problem characteristic set domain.
- Build AS-Micro-Array. The AS-Micro-array is created considering the domain found and the list of problem characteristics. It represents the application specification.

- MM Selection. The MM selection is performed employing the AS-Micro-Array as selection criteria. The selection is accomplished using a knowledge base of MM characterizations.
  - Select MM Micro-Arrays. The matching process is performed evaluating the AS-Micro-Array with each MM-Micro-Array of the Knowledge Base. At the end the set of best MM-Micro-Arrays are selected as candidates.
  - Create MM Agents. The agents are built using each MM-Micro-Array candidate and a copy of the AS-Micro-Array. Each agent owns two main behavior tasks: Negotiation and Selection.

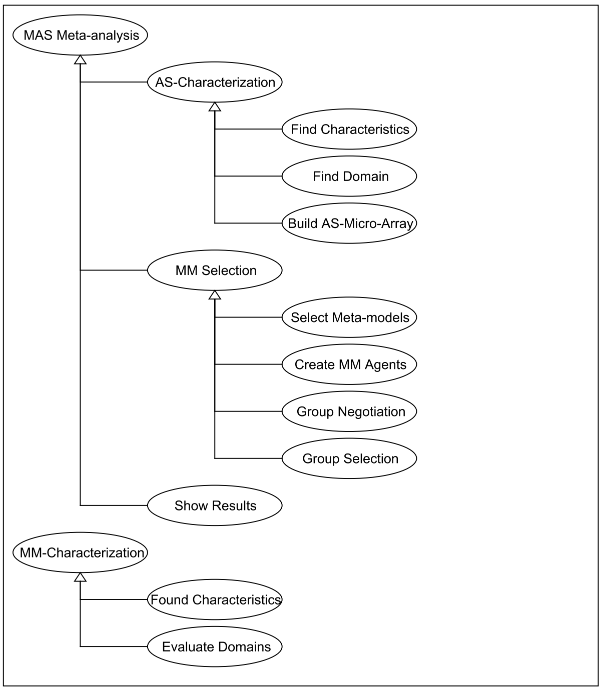


Figure 7.4: System use cases

  - Group Negotiation. Each agent must build a solution group negotiating the membership with the other agents. It is done considering only the different than the owned MM types and evaluating the compatibility between MM-Micro-Arrays. This process finish when all the agents have been contacted between all or when the execution time runs out.
  - Group Selection. When each agent has a solution group the results of each solution group are shared between all the agents. Each agent compares the compatibility of all the solutions group with the AS-Micro-array in order to select the best one. Finally each agent votes for the winning group sending his vote to the observer.

- Show Results. The results are shown in a window considering the agent votes. Thus the winning solution group is revealed. If there is a tie between two or more groups the system cannot provide a winning group.

7.1.3 MM-Characterization

Abstract from an existing meta-model the required information to characterize it into a MM-Micro-Array.

- Found Features. Describe the Meta-model through features within a class.
- Evaluate Domains. Add the success statistical values of each feature within a domain of solution in order to locate the meta-model as a successful component for a given domain.

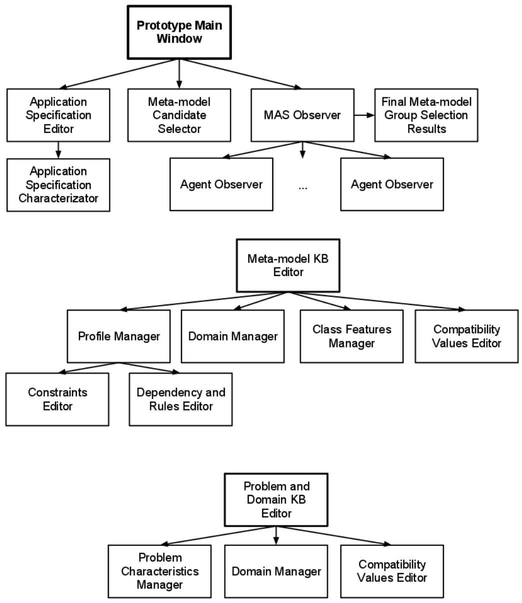


Figure 7.5: System Navigation Diagram

7.1.4 Navigation Diagram

The prototype is structured as follows in the navigation diagram displayed into the figure 7.5. It is composed of three parts where the main part is the Prototype Main window the other two parts are the knowledge bases editors for characterizations. The Prototype Main Window allows to go through the AS-Characterization, MM-Candidate Selector and MAS-Observer windows in order to follow and perform the approach proposed process.

7.1.5 Class diagrams

The class diagrams defined for the agent classes are shown in the fig. 7.6

In the following sections we will present each part of the tool.

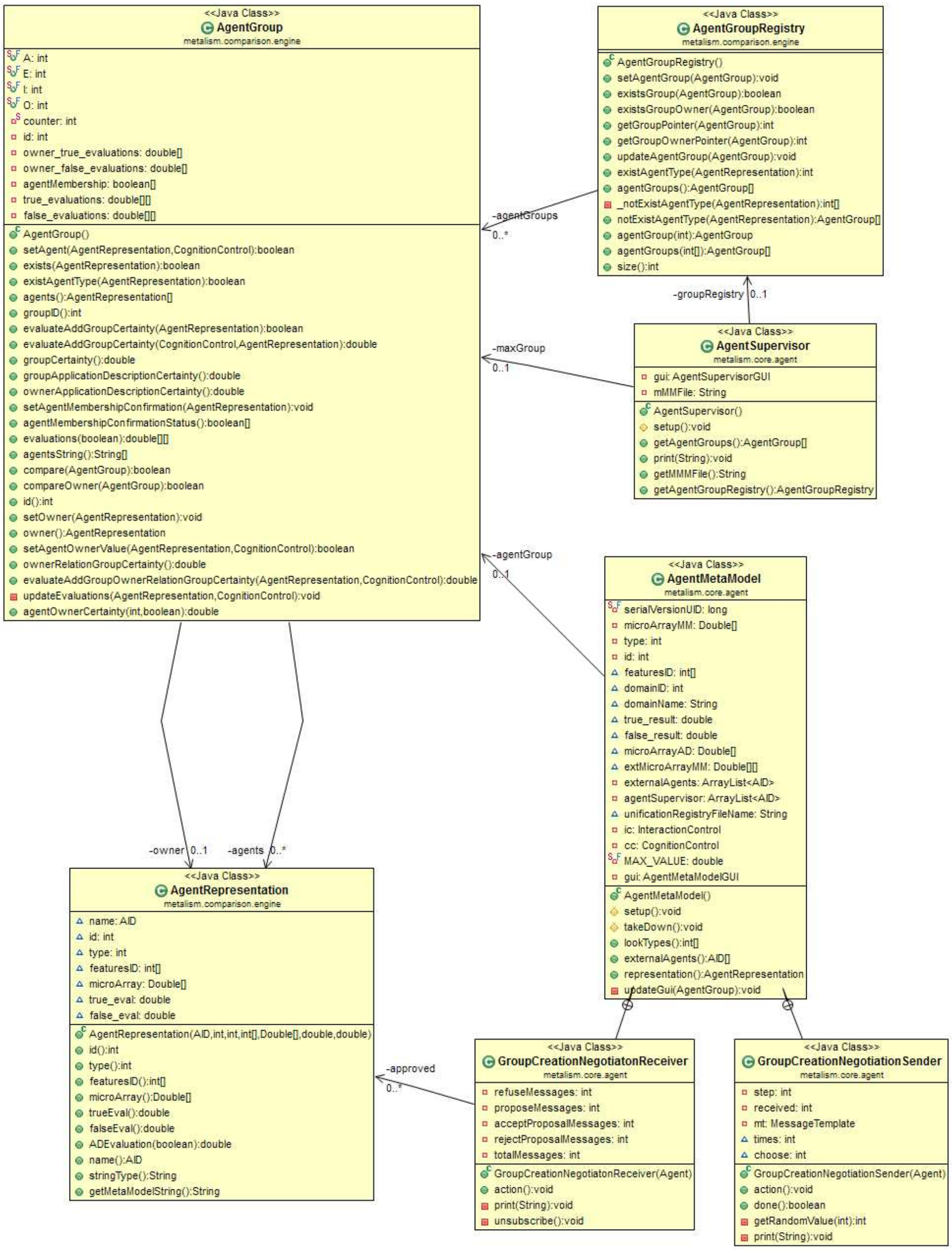


Figure 7.6: Agent Class diagram

Figure 7.7: Agent Cognition Class diagram

7.2 Graphic User Interface (GUI)

The GUI is composed by 2 different modules:

- Knowledge base creator. This module has two submodules:
  - Domain-Problem Editor. That permits to create KB's of domains and problem characteristics (See fig. 7.8).
  - Meta-model Manager. It allows to define meta-model features and therefore create meta-model using such features (See fig. 7.9)
- Matching Engine. Is the tool that allows to perform the matching process (See fig. 7.10), however, it is composed by the next submodules:
  - KB Loaders. There are two buttons that allows to load a Domain-Problem KB and Meta-model KB in order to perform the matching process.
  - Domain Selector. It allows to select a domain according to the data contained in the micro-array (See fig.7.11). This can be selected manually or obtained using the approach defined in Appendix B.
  - Meta-model candidate chooser. Using the Meta-models KB data, the selected domain and the micro-array as selection criteria this module allows to choose the candidates subset of meta-models from the KB (See figs. 7.127.13).
  - Matching viewer. When the meta-model candidates have been selected and the start engine button is pressed the matching engine starts and then the matching viewer module displays the current state of the agents that own the MAS meta-models (See. fig. 7.14). It shows the agent data about the meta-model owned and the currently solution group built. At the end of the process is possible to display a chart showing the results of the groups to make evident the best group considering the probability of success (See fig. 7.15).

In the next sections we briefly show screens of each part of the software prototype explaining their work.

7.2.1 Knowledge bases

The Knowledge bases can be created from scratch or employing an automated approach similar as (Sánchez and Moreno, 2008) to create Domains, AS-characteristics and MM-features KB through the web or document analysis. As we stated in section 5.3 we consider it an issue that is not the main focus of this thesis. So, we have implemented these software modules to create manually such KB.

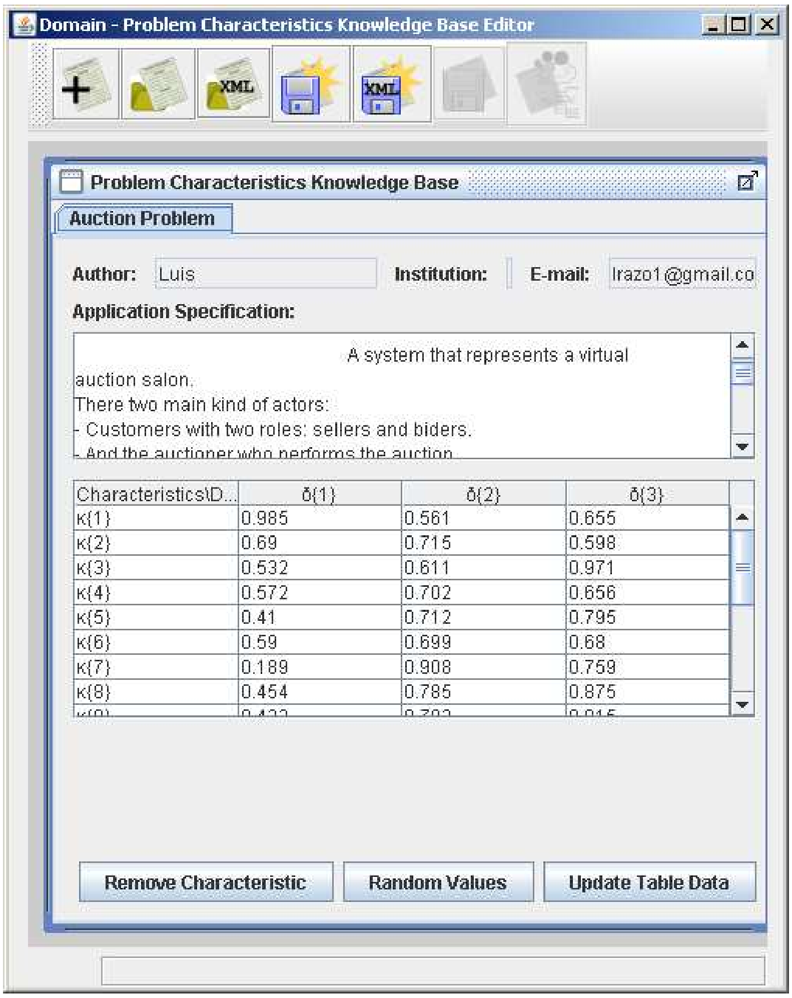


Figure 7.8: Domain - AS Characteristics GUI.

7.2.1.1 Domain-AS Characteristics

Therefore in the fig. 7.8 we can add characteristics and domains, for each relation "‘as-characteristic - domain"’ we must define a value. However when we add a new domain or a new as-characteristic the default value is 0.5 that means the maximal uncertainty or entropy.

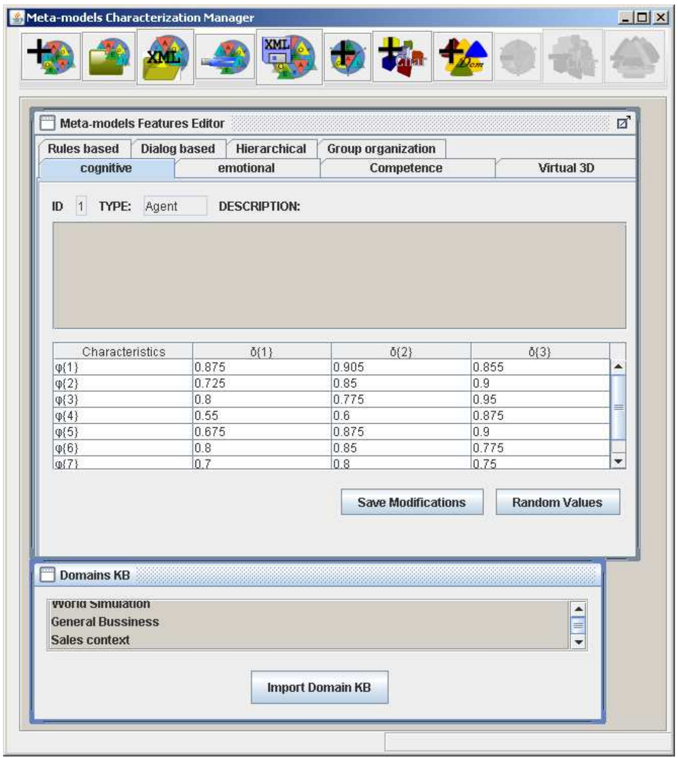


Figure 7.9: Meta-model Features GUI

7.2.1.2 Meta-model features

We can also edit and create meta-models and their features as we can see in the fig. 7.9.

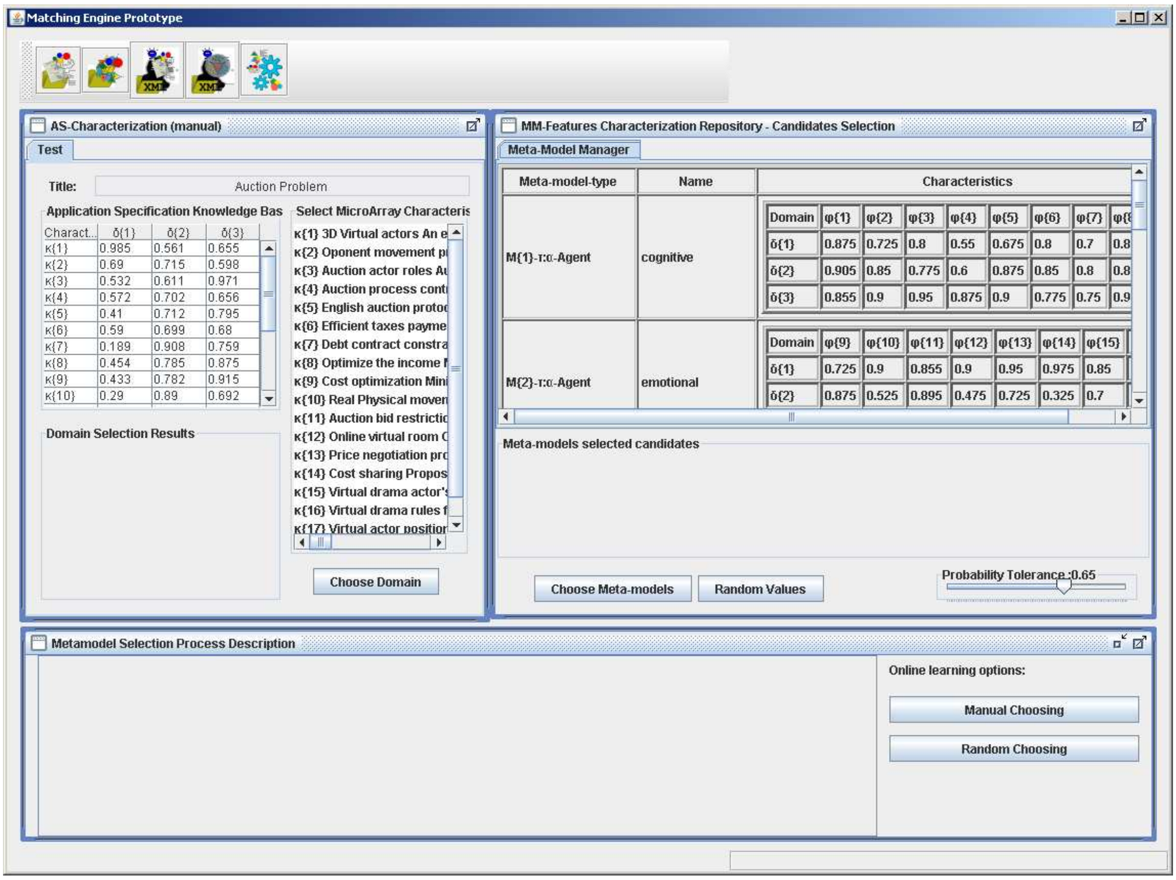


Figure 7.10: Matching Engine Overview

7.2.2 Matching Engine

Moreover to perform the match task we have created the Matching Engine module, the GUI is showed in fig. 7.10, where we load the KB created using the Editor showed above.

The matching engine GUI is divided in three parts:

- AS-Characterization
- MM-Features Characterization
- Meta-model selection process

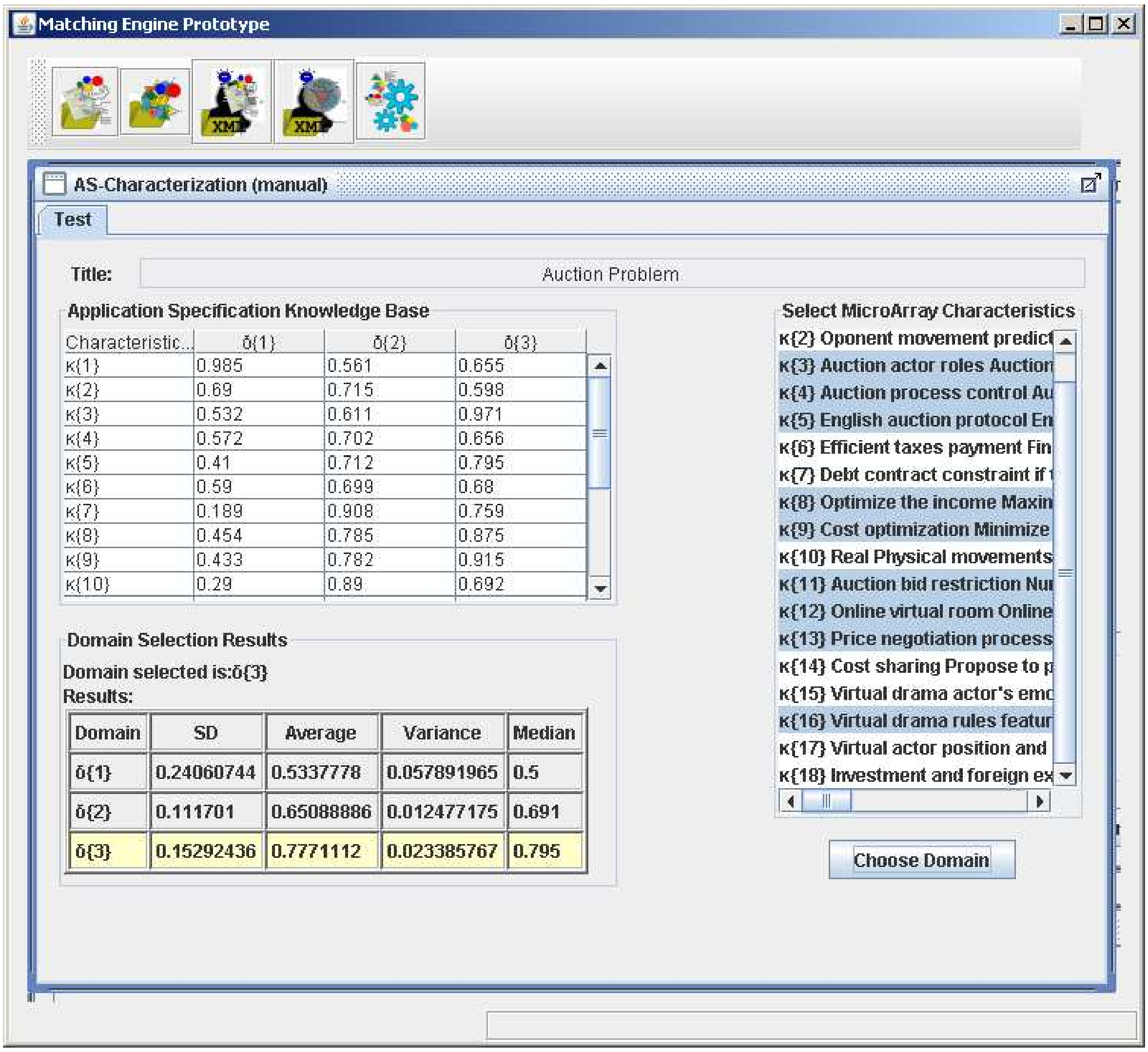


Figure 7.11: AS-Characteristics GUI from Matching Engine Prototype.

7.2.2.1 AS-Characterization

In this part we visualize the AS-Characteristics KB and we can manually perform the characterization selecting a set of available characteristics thus considering such a set automatically select the AS belonging domain (See fig. 7.11). However we can load automatically characterized values from a previously characterized micro-array file.

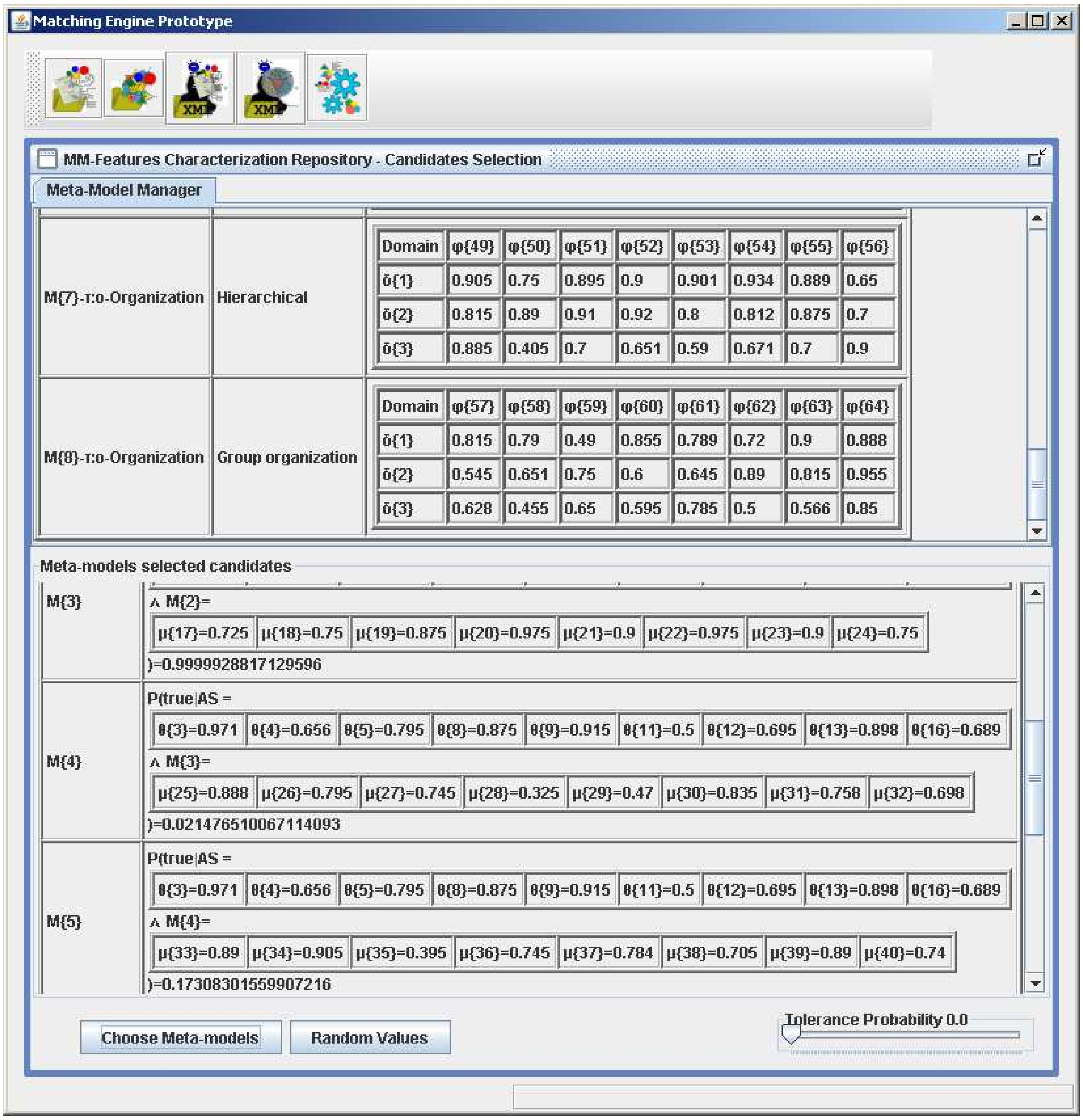


Figure 7.12: MM-Features GUI

7.2.2.2 Candidate Selector

Having found the domain through the characterization the next step is to find the candidate meta-models to be used in the construction of the solution (See fig. 7.12). For that uses the knowledge base that contains the meta-models and micro-array and domain obtained in the previous phase.

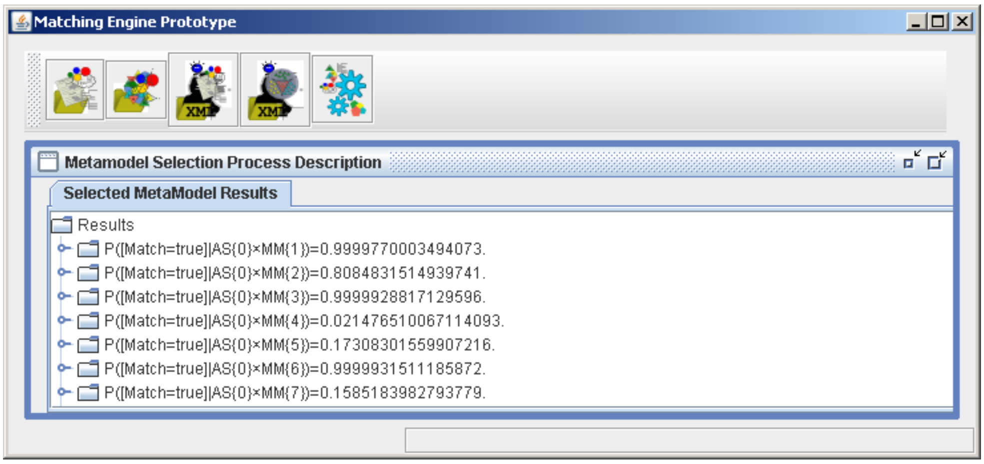


Figure 7.13: MM-Features GUI

The selection process is detailed in the section 6.5.3 and exemplified in the section 8.3.0.3. The figure 7.13 discloses the prototype results of such a selection process.

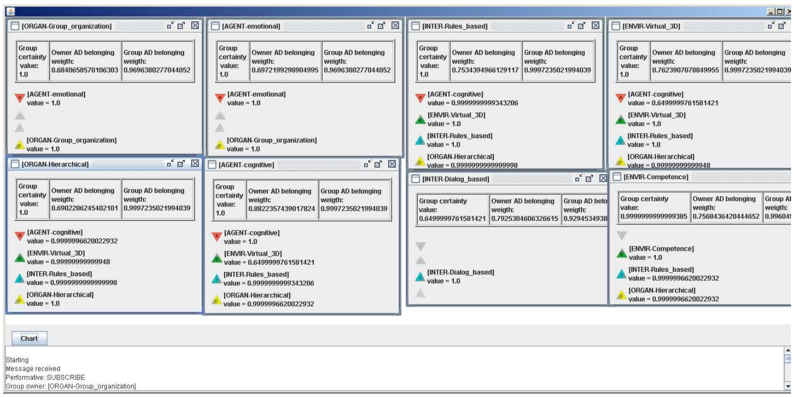


Figure 7.14: MAS Observers that shows the inner state of each agent.

7.2.3 MAS Observation

7.2.3.1 MAS Decision society

As we stated in the section I we propose to employ a society of agents that make decisions individually and collectively. Therefore we can see in the fig. 7.14 the MAS Observers showing the inner state of each agent while they interact between them to build the solution groups. Such agent's inner process is detailed in 6.5.4 moreover an example is showed in 8.4.

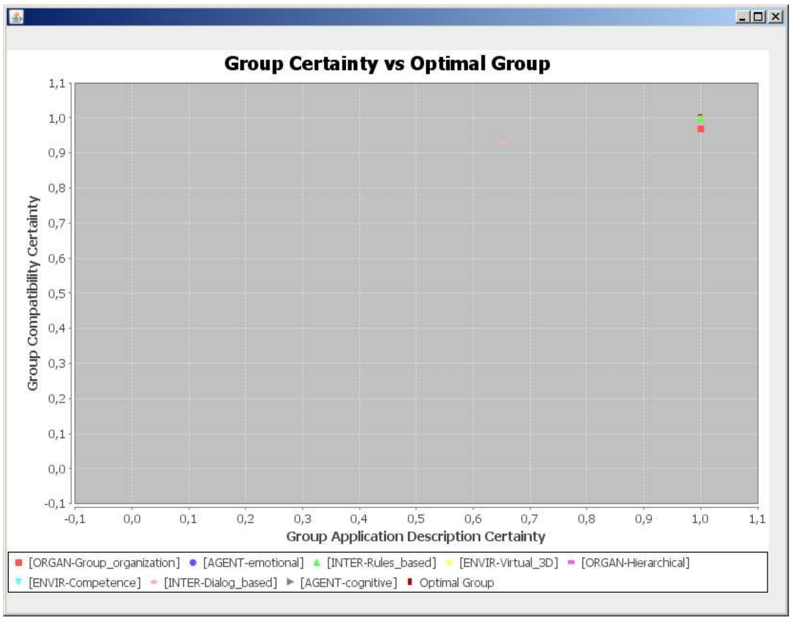


Figure 7.15: Group solutions chart

7.2.3.2 Chart

Finally we can observe the groups chart considering the results of the agent's performance. We can see the optimal or expected group values against the resulting groups. We consider two axis: Certainty about the "'Application Specification (or Description)"' and group compatibility certainty. The highest likelihood in both axes is nearest to 1.0, the lowest is nearest to 0.0 (See fig. 7.15).

Chapter 8

Case Study and Validation

To evaluate our approach we have created the following case study to test it. We present, along each stage of the proposed approach, an example that demonstrates the operation of our approach stage by stage. So we start with the AS-Characterization stage example in the next lines.

8.1 Application Specification

The present case study analyzes a system that requires to create a virtual auction room that is textually described as follows:

> A system that represents a virtual auction salon. There two main kinds of actors:
>
> - Customers with two roles: sellers and bidders.
> - And the auctioneer who performs the auction following the rules specified.
>
> Both customers actors, buyers and bidders, can create their representative customizing a behavior as they want, a virtual entity that represents the customer in the virtual auction salon. These virtual entity has a customizable behavior it means that the real customer can add the following features:
>
> - A maximum budget for a list or category of articles.
> - Also they could accept rules to make or not bids.
>
> In the other side the virtual salon will be managed by an automated virtual entity that will create auctions taking proposed goods to sell from the customers that want to sell in the auction. So we will accept goods with an initial value and a desired selling value so we will add a 20% of management costs to the final selling value. Every auction could be customizable in the way that the selling customer could chose between four main types of auction rules:

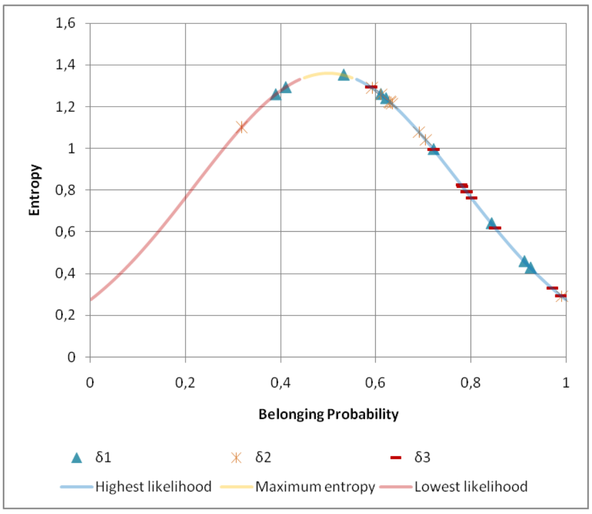


Figure 8.1: Automated Auction AS characterization chart

- English auction.
- Dutch auction.
- Sealed first-price auction.
- Vickrey auction.

Therefore the process is described in the following sections.

8.2 AS-Characterization Example

Usually the both KB - C_{KB} and D_{KB} - employed in the AS-Characterization process have a large number of AS-characteristics and domains respectively. However, for reasons of space the number of AS-characteristics and domains employed in the following example stages have been reduced and they are not exhaustive.

The AS characterized in the example is about an application that desires an automated auction system - we will refer to it as AS_{AA} -. The AS-characterization process approach employed is based on textual AS-characteristics. It employs text-based pattern recognition - similar to the one employed in a statistical machine translation(Koehn, 2009) - and uses regular expressions - similar as described in (Bilisoly, 2008) -. However, the AS characterization process could be implemented differently using other approaches, nevertheless, such characterizations processes are not the goal of this paper. Thus the AS-characterization stage example is illustrated in the following pages:

The KB of domains and characteristics employed are as follows:

D_{KB}	
Domain	Description
δ_1	Simulation domain.
δ_2	Negotiation based domain.
δ_3	Public sales context domain.

C_{KB}	
Characteristic	Description
κ_1	an environment with 3d virtual actors .
κ_2	analyze the opponent and predict his movements or decisions.
κ_3	auction actors are customers with buyer and seller roles.
κ_4	auctioneer actor controls the auction process.
κ_5	English auction protocol.
κ_6	find the best hints to reduce the taxes payment.
κ_7	if the debtor fails to pay the debt according to the contract shall be entitled to a lien.
κ_8	maximize the income or gain.
κ_9	minimize the cost or price.
κ_{10}	movements should be considered as similar to the real physical.
κ_{11}	number of bids limited.
κ_{12}	on-line virtual room.
κ_{13}	price negotiation process.
κ_{14}	propose to pay costs and sharing equally.
κ_{15}	the actors express their emotions with facial expressions and body postures.
κ_{16}	the actors must follow the defined rules.
κ_{17}	the actors will learn to manage their postures using their experience.
κ_{18}	the investment considers the foreign exchange rates.

These domains and characteristics were defined inspired in the work described in (Milidiu et al., 2003) and (Zúñiga et al., 2005) but simplified to be showed here. Moreover the AS-characterization KB has the following values:

AS_{KB}									
	Characteristics								
Domain	κ_1	κ_2	κ_3	κ_4	κ_5	κ_6	κ_7	κ_8	κ_9
δ_1	0,985	0,69	0,532	0,572	0,41	0,59	0,189	0,454	0,433
δ_2	0,561	0,715	0,611	0,702	0,712	0,699	0,908	0,785	0,782
δ_3	0,655	0,598	0,971	0,656	0,795	0,68	0,759	0,875	0,915
Domain	κ_{10}	κ_{11}	κ_{12}	κ_{13}	κ_{14}	κ_{15}	κ_{16}	κ_{17}	κ_{18}
δ_1	0,805	0,29	0,893	0,119	0,201	0,944	0,891	0,864	0,39
δ_2	0,302	0,89	0,48	0,691	0,854	0,302	0,595	0,523	0,823
δ_3	0,194	0,692	0,695	0,898	0,876	0,389	0,689	0,37	0,705

The characterization process only chooses the features that are detected within the analyzed AS, then when the characterization process has finished we have the next result:

AS'_{AAS}									
	Characteristics found								
Domain	κ_3	κ_4	κ_5	κ_8	κ_9	κ_{11}	κ_{12}	κ_{13}	κ_{16}
δ_1	0,532	0,572	0,410	0,454	0,433	0,290	0,893	0,119	0,891
δ_2	0,611	0,702	0,712	0,785	0,782	0,890	0,480	0,691	0,595
δ_3	0,971	0,656	0,795	0,875	0,915	0,692	0,695	0,898	0,689

	Average	Std. Dev.	Variance	Median
δ_1	0,510	0,254	0,065	0,454
δ_2	0,694	0,121	0,015	0,702
δ_3	0,798	0,119	0,014	0,795

The chart - see fig. 8.1 - shows the AS-characteristics with their belonging domains values adjusted to a gauss bell. The characteristics tendency shows that the domain with highest likelihood is the δ_3 because all their characteristics values are in the highest likelihood area. Also the δ_3 has the lowest variance and standard deviation, however, the $\delta_3$median is the highest between the other domains. So, this mean that the AS-characterization reduces the uncertainty about the AS belonging domain through the characteristics localization. Thus the final result is as follows:

AS'_{AAS} (Micro-array)									
	Characteristics found								
Domain	κ_3	κ_4	κ_5	κ_8	κ_9	κ_{11}	κ_{12}	κ_{13}	κ_{16}
δ_3	0,971	0,656	0,795	0,875	0,915	0,692	0,695	0,898	0,689

In this basic case where only three domains are used we choose only one - the most adapted -, however, it is noted that is possible to choose more than one domain.

8.3 MM-Characterization Example

In the context of the AS_{AAS} characterization example - started in the section 8.2 and considering the same domains δ_1, δ_2 and δ_3 - we employ the following MM-types:

Agents:	α_1:Cognitive Agent	α_2:Emotional Agent
Environments:	ε_1:Competence Env.	ε_2:Virtual 3D Env.
Interactions:	ι_1:Rule-based Int.	ι_2:Dialog-based Int.
Organizations:	o_1:Hierarchical Org.	o_2:Group Org.

In order to obtain all the Meta-models characterization values - consequently add them to the KB - the MM were first analyzed from the literature, therefore, the MM-features were abstracted from each MM and defined using text sentences (see table 8.1). The percentage efficacy values were obtained by considering the information provided through oral discussion with our laboratories experienced members (see table 8.2). However, this process can be automated using a knowledge engineering and pattern recognition approach, but this process is beyond of the scope of this thesis. For the cognitive agent and the distributed semi-observable environment the features are defined using the notation of (Stuart Russell, 2009), and for the emotional agent and virtual 3D environment the features described in (Ramos et al., 2005). The rule based interactions are similar to the one discussed in (Uckelman, 2010) and dialog-based interactions features were abstracted from (Parsons et al., 2002).
The hierarchical organization features were inspired in (Brahmi and Gammoudi, 2009) and the group organization in the contract-net protocol (FIPA, 2002). The meta-model features are as the following extract shows:

Therefore we employ a MM-Features efficacy values KB as the extract of table 8.2 shows. Thus, defining together both KB: MM-Features and efficacy values; we have the basic data needed as "fuel" for the comparison engine that we will explain in detail in the following section.

Table 8.1: MM-features extract

φ_1	deliberative cognition	φ_{32}	fully-observable
φ_2	knowledge representation about itself and others	φ_{33}	basic rule language
		φ_{34}	communication timeouts
φ_3	strategic competence	φ_{35}	delimited messages
φ_4	independent performance	φ_{36}	broadcast communication
φ_5	cooperative behavior	φ_{37}	meeting messages
φ_6	uses knowledge about organization	φ_{38}	non response penalization
φ_7	social listening to others	φ_{39}	answer-response sequences
φ_8	tactic actions and decisions	φ_{40}	knowledge rule transmission
φ_9	third agents representation	φ_{41}	minimal set of words
φ_{10}	emotional cognition	φ_{42}	dialog timeout
φ_{11}	goal based behavior	φ_{43}	multiple listeners dialog
φ_{12}	agent interaction through sensors and effectors	φ_{44}	idea storm kind dialog
		φ_{45}	emotional context features dialog
φ_{13}	agent environment interaction sensibility		
		φ_{46}	observable dialog feature
φ_{14}	drama simulation skill	φ_{47}	secret dialog feature
φ_{15}	dialog-based skill	φ_{48}	dialogical knowledge share
φ_{16}	dynamic goal generation skill	φ_{49}	hierarchy defined rules
φ_{17}	partially observable	φ_{50}	hierarchy levels
φ_{18}	deterministic	φ_{51}	hierarchy allowed actions
φ_{19}	sequential actions	φ_{52}	hierarchy roles
φ_{20}	dynamic changes	φ_{53}	hierarchy same level groups
φ_{21}	continuous time line	φ_{54}	hierarchy upgrade level rules
φ_{22}	multi-agent support	φ_{55}	communication permissions
φ_{23}	competitive rules	φ_{56}	service provider reference
φ_{24}	multi-location	φ_{57}	membership rules
φ_{25}	stochastic	φ_{58}	group registry
φ_{26}	episodic	φ_{59}	group evaluation
φ_{27}	semi-dynamic	φ_{60}	group knowledge
φ_{28}	distributed visual port	φ_{61}	group constraints
φ_{29}	physical interaction simulation	φ_{62}	membership protocol
φ_{30}	pre-conditions and post-conditions	φ_{63}	group members roles
φ_{31}	normative	φ_{64}	group goals

Table 8.2: MM-features efficacy values KB extract

Domains	Features							
$\tau_1 : \alpha_1$	φ_1	φ_2	φ_3	φ_4	φ_5	φ_6	φ_7	φ_8
δ_1	0,875	0,725	0,8	0,55	0,675	0,8	0,7	0,825
δ_2	0,905	0,85	0,775	0,6	0,875	0,85	0,8	0,85
δ_3	0,855	0,9	0,95	0,875	0,9	0,775	0,75	0,9
$\tau_2 : \alpha_2$	φ_9	φ_{10}	φ_{11}	φ_{12}	φ_{13}	φ_{14}	φ_{15}	φ_{16}
δ_1	0,725	0,9	0,855	0,9	0,95	0,975	0,85	0,85
δ_2	0,875	0,525	0,895	0,475	0,725	0,325	0,7	0,8
δ_3	0,695	0,4	0,7	0,525	0,625	0,375	0,725	0,825
$\tau_3 : \varepsilon_1$	φ_{17}	φ_{18}	φ_{19}	φ_{20}	φ_{21}	φ_{22}	φ_{23}	φ_{24}
δ_1	0,9	0,625	0,875	0,9	0,35	0,95	0,9	0,725
δ_2	0,6	0,7	0,8	0,925	0,8	0,975	0,875	0,6
δ_3	0,725	0,75	0,875	0,975	0,9	0,975	0,9	0,75
$\tau_4 : \varepsilon_2$	φ_{25}	φ_{26}	φ_{27}	φ_{28}	φ_{29}	φ_{30}	φ_{31}	φ_{32}
δ_1	0,785	0,725	0,92	0,935	0,965	0,94	0,745	0,891
δ_2	0,685	0,725	0,824	0,451	0,395	0,855	0,815	0,785
δ_3	0,888	0,795	0,745	0,325	0,47	0,835	0,758	0,698
$\tau_5 : \iota_1$	φ_{33}	φ_{34}	φ_{35}	φ_{36}	φ_{37}	φ_{38}	φ_{39}	φ_{40}
δ_1	0,95	0,745	0,565	0,855	0,69	0,905	0,72	0,9
δ_2	0,885	0,955	0,435	0,745	0,895	0,645	0,91	0,89
δ_3	0,89	0,905	0,395	0,745	0,784	0,705	0,89	0,74
$\tau_6 : \iota_2$	φ_{41}	φ_{42}	φ_{43}	φ_{44}	φ_{45}	φ_{46}	φ_{47}	φ_{48}
δ_1	0,72	0,812	0,86	0,88	0,86	0,91	0,78	0,61
δ_2	0,92	0,95	0,901	0,25	0,56	0,89	0,77	0,7
δ_3	0,75	0,925	0,68	0,45	0,61	0,78	0,79	0,65
$\tau_7 : o_1$	φ_{49}	φ_{50}	φ_{51}	φ_{52}	φ_{53}	φ_{54}	φ_{55}	φ_{56}
δ_1	0,905	0,75	0,895	0,9	0,901	0,934	0,889	0,65
δ_2	0,815	0,89	0,91	0,92	0,8	0,812	0,875	0,7
δ_3	0,885	0,405	0,7	0,651	0,59	0,671	0,7	0,9
$\tau_8 : o_2$	φ_{57}	φ_{58}	φ_{59}	φ_{60}	φ_{61}	φ_{62}	φ_{63}	φ_{64}
δ_1	0,815	0,79	0,49	0,855	0,789	0,72	0,9	0,888
δ_2	0,545	0,651	0,75	0,6	0,645	0,89	0,815	0,955
δ_3	0,628	0,455	0,65	0,595	0,785	0,5	0,566	0,85

8.3.0.3 Matching Example

We retake the context of the sections 8.2 and 8.3 to continue the example. Now we can ask which is the probability to successful match a MM-characterization as part of the solution of the AS-characterization in the next way:

AS'_{AAS} (Micro-array)									
	Characteristics found								
Domain	θ_3	θ_4	θ_5	θ_8	θ_9	θ_{11}	θ_{12}	θ_{13}	θ_{16}
δ_3	0,971	0,656	0,795	0,875	0,915	0,692	0,695	0,898	0,689
Domain	Equivalent boolean values								
δ_3	$true$	$false$	$true$	$true$	$true$	$false$	$false$	$true$	$false$

MM'_1								
Domain	Features							
$\tau_1 : \alpha_1$	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8
δ_3	0,855	0,9	0,95	0,875	0,9	0,775	0,75	0,9
Domain	Equivalent boolean values							
δ_3	$true$	$true$	$true$	$true$	$true$	$true$	$true$	$true$

Therefore we have:

$$AS'_{AAS} = \{[\theta_3 = true], [\theta_4 = false], [\theta_5 = true], [\theta_8 = true], [\theta_9 = true], [\theta_{11} = false], [\theta_{12} = false], [\theta_{13} = true], [\theta_{16} = false]\}$$

$$MM'_1 = \{[\mu_1 = true], [\mu_2 = true], [\mu_3 = true], [\mu_4 = true], [\mu_5 = true], [\mu_6 = true], [\mu_7 = true], [\mu_8 = true]\}$$

$$L_{AAS,1} = AS'_{AAS} \times MM'_1$$

$L_{AAS,1}$(fragment)									
(θ_i, μ_k)	$\theta_3 = t$	$\theta_4 = f$	$\theta_5 = t$	$\theta_8 = t$	$\theta_9 = t$	$\theta_{11} = f$	$\theta_{12} = f$	$\theta_{13} = t$	$\theta_{16} = f$
$\mu_1 = t$	(t)	(f)	(t)	(t)	(t)	(f)	(f)	(t)	(f)
$\mu_2 = t$	(t)	(f)	(t)	(t)	(t)	(f)	(f)	(t)	(f)
$\vdots$	$\vdots$								
$\mu_8 = t$	(t)	(f)	(t)	(t)	(t)	(f)	(f)	(t)	(f)

Considering the outcome derived from 20 surveys where the matched were 15, and the unmatched were 5 we have the following histogram fragment:

Histogram (δ_3)			
j	$\lambda_j = (\theta_i, \mu_k)$	n_f^j	n_t^j
129	(θ_3, μ_1)	0	9
193	(θ_4, μ_1)	5	14
257	(θ_5, μ_1)	4	15
449	(θ_8, μ_1)	5	14
513	(θ_9, μ_1)	5	0
641	(θ_{11}, μ_1)	0	12
705	(θ_{12}, μ_1)	0	12
833	(θ_{13}, μ_1)	5	14
961	(θ_{16}, μ_1)	5	13
$\vdots$	$\vdots$		

Therefore we obtain the next evaluation results (fragment):

$P([\,Match\ =\ true\,]\ \vert\ (\theta_i, \mu_k)\ )$									
(θ_i, μ_k)	$\theta_3 = t$	$\theta_4 = f$	$\theta_5 = t$	$\theta_8 = t$	$\theta_9 = t$	$\theta_{11} = f$	$\theta_{12} = f$	$\theta_{13} = t$	$\theta_{16} = f$
$\mu_1 = t$	0.588	0.882	0.941	0.882	0.059	0.765	0.765	0.882	0.824
$\vdots$	$\vdots$								

Therefore answering the question (using the expressions from the section 6.5.3.2):

$$P(\ [Match = true] \mid \lambda_{129} \wedge \ldots \wedge \lambda_{968}) = 0.9902$$

The result indicates that, based on experience lodged in the histogram, the probability of compatibility of the meta-model α_1 with the AS-description is high. Thus this meta-model is chosen as a candidate and create an agent to represent the meta-model.

8.4 Solution Example

To follow the example, consider a couple of agents carrying the Meta-models of cognitive agent and competitive environment, α_1 and ε_1 respectively, from the table 8.1.

Therefore following a similar path as showed in 8.3.0.3 we answer the question from the section 6.5.4.3:

$$P(\ [Solution = true] \mid \sigma_1 \wedge \ldots \wedge \sigma_8) = 0.9099$$

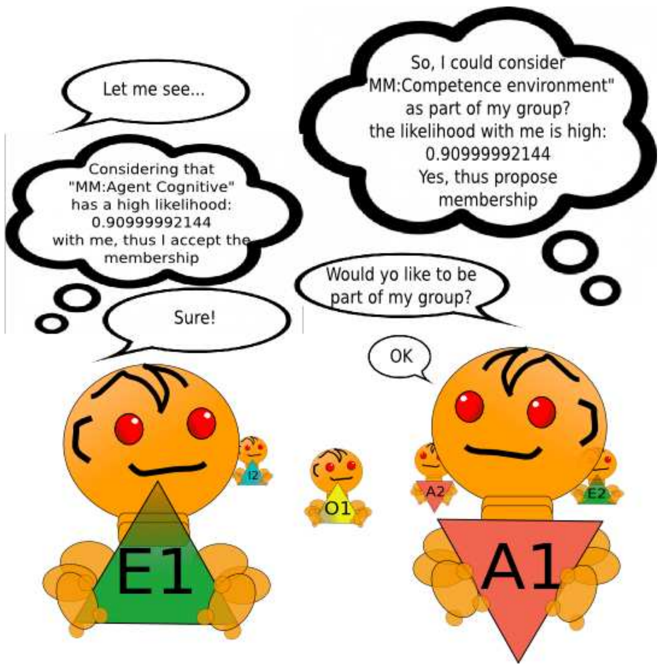


Figure 8.2: Agent interaction recreation

Thus the result indicates that the compatibility likelihood of the meta-models α_1 and ε_1 is high (near to 1), then, the agent that represents α_1 adds to his solution group the represented meta-model ε_1.

8.5 Matching Example

Furthermore for space reasons in the example shown previously we demonstrate some fragments of the Bayesian algorithm that are employed within the decision-making process. However our main objective is to show the core inner process of making decision. Such process is performed at each decision within the cognition module within each agent in the system. So, taking into account such performance we could explain the entire process at agent behavior and interaction high level.

As we have shown in section 8.2 for the auction problem we establish simply 9 problem characteristics - connected to such case - and the problem domain was predominantly the defined as *"‘δ_3: Public sales context"’*.

At once 6 MM-characterizations were chosen as candidates from the MM-characterization KB. For each MM-characterization is created an agent, thus they made up 6 different groups.

Thus the agents interact between them to find the best group partners evaluating the probability of compatibility of each possible option (See fig. 8.2). At the final step the agents chosen between them the best group created by considering the highest values from the group's members compatibility and the group's compatibility with the problem.

Lastly the most compatible group found in our example was the formed by the meta-models:

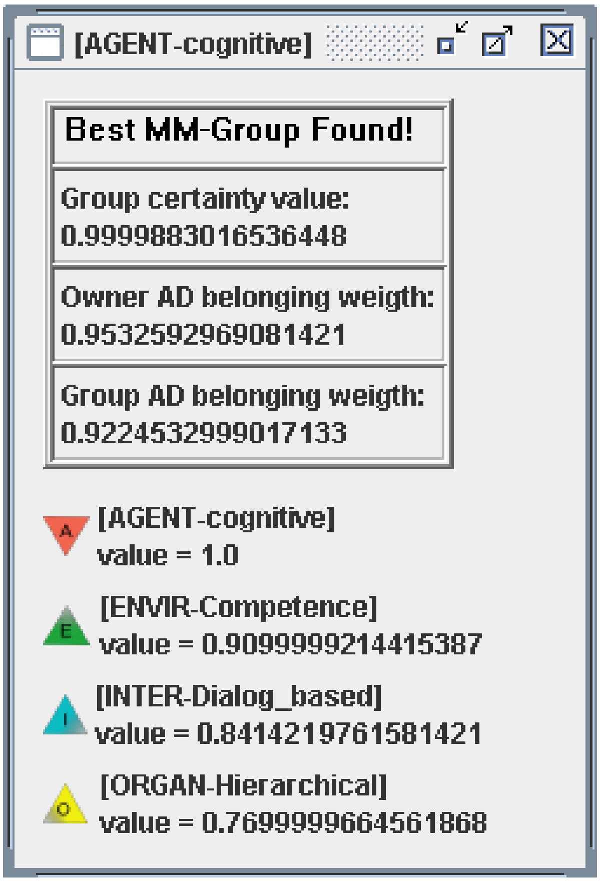


Figure 8.3: Best compatible meta-models group found in the case study test.

cognitive agent, competence environment, dialog-based interaction and hierarchical organization (See fig. 8.3).

8.6 Profiling Results

Analyzing the performance of the tool we have employed the Netbeans profiler to find out which are the most significative operations and the time required to perform them. The term profile within the developing context means a way to analyze the performance of a program or gain a comprehensive understanding of the overall performance of an application. In our case we are interested in analyzing the execution time of the agents interactions in order to evaluate the tool overall performance.

In the figure 8.4 we show the results of the overall project execution where each method is evaluated considering the number of calls and the time required to perform such calls. As we can see in the figure the most called methods are the related with the read/write of the unification registry. It means that the agents where constantly employing such registry, that stores the unification knowledge base, in order to evaluate the inter meta-models compatibilities. The next top list methods ‘”value” and “compare” are related to such compatibilities evaluations.

Hot Spots - Method	Self time [%]	Self time	Invocations
metalism.core.util.CognitionControl.**openUnificationRegistry** ()		130837 ... (55,1%)	8
metalism.core.util.UnificationMatrix.**value** (int, int)		18487 ms (7,8%)	78848
metalism.core.util.UnificationValue.**compare** (int, int)		14191 ms (6%)	2562560
org.jdom.Verifier.**isXMLNameCharacter** (char)		3966 ms (1,7%)	4297583
metalism.core.agent.gui.AgentGroupReporter.**initAgent** ()		3900 ms (1,6%)	864
metalism.core.agent.gui.AgentGroupReporter.**initComponents** ()		2615 ms (1,1%)	864
org.jdom.Verifier.**checkXMLName** (String)		2533 ms (1,1%)	766214
org.jdom.Verifier.**isXMLLetter** (char)		2336 ms (1%)	5063797
metalism.core.agent.gui.AgentSupervisorGUI.**initInternalFrames** ()		2173 ms (0,9%)	113
metalism.core.agent.gui.Usher.**componentAdded** (java.awt.event.ContainerEvent)		1956 ms (0,8%)	51374
org.jdom.Verifier.**checkCharacterData** (String)		1684 ms (0,7%)	575219
org.jdom.AttributeList.**indexOf** (String, org.jdom.Namespace)		1452 ms (0,6%)	1274971
metalism.core.util.CognitionControl.**P** (boolean, int, int, int[], int[], Double[], Double[], metalism.core.util.UnificationMatrix)		1334 ms (0,6%)	1016
org.jdom.Attribute.**<init>** (String, String, int, org.jdom.Namespace)		1179 ms (0,5%)	574943
org.jdom.Verifier.**isHighSurrogate** (char)		1083 ms (0,5%)	2689555
metalism.core.util.RelationsGraph.**init** ()		1049 ms (0,4%)	8
org.jgraph.graph.DefaultCellViewFactory.**createVertexView** (Object)		1038 ms (0,4%)	40
org.jdom.AttributeList.**add** (int, org.jdom.Attribute)		972 ms (0,4%)	574943
metalism.core.util.UnificationValue.**n** (int, int)		866 ms (0,4%)	138752
metalism.comparison.engine.EngineMainWindow.**initComponents** ()		819 ms (0,3%)	1
metalism.core.util.CognitionControl.**finalP** (boolean, Double, Double, metalism.core.util.UnificationValue)		785 ms (0,3%)	65024
org.jdom.output.Format$DefaultEscapeStrategy.**shouldEscape** (char)		783 ms (0,3%)	935792
org.jdom.Attribute.**getNamespaceURI** ()		780 ms (0,3%)	1907875
org.jdom.input.SAXHandler.**startElement** (String, String, String, org.xml.sax.Attributes)		773 ms (0,3%)	89107
org.jdom.Verifier.**isXMLNameStartCharacter** (char)		770 ms (0,3%)	766214
metalism.core.util.UnificationValue.**totalGlobal** (int)		760 ms (0,3%)	65024
org.jdom.AttributeList.**add** (Object)		738 ms (0,3%)	574943
metalism.core.agent.gui.Usher.**descend** (boolean[][], java.awt.Dimension, java.awt.Component[], java.awt.Component, java.awt.Point)		719 ms (0,3%)	410306
org.jdom.Element.**<init>** (String, org.jdom.Namespace)		709 ms (0,3%)	191271
org.jdom.Verifier.**isXMLCharacter** (int)		703 ms (0,3%)	1753763
org.jdom.output.XMLOutputter.**printAttributes** (java.io.Writer, java.util.List, org.jdom.Element, org.jdom.output.XMLOutputter.NamespaceStack)		694 ms (0,3%)	102164
org.jdom.input.SAXBuilder.**build** (org.xml.sax.InputSource)		687 ms (0,3%)	2
metalism.comparison.engine.Boot.**writeUnificationRegistryXMLFile** (String, metalism.core.util.UnificationRegistry)		634 ms (0,3%)	2
jade.lang.acl.ACLMessage.**getContentObject** ()		618 ms (0,3%)	264
org.jdom.Element.**setAttribute** (String, String)		610 ms (0,3%)	307248

Figure 8.4: Profiling Overall Project

Call Tree - Method	Time [%]	Time	Invocations
AWT-EventQueue-0		41772 ms (100%)	1
MMAGENT[AGENT-emotional]		22910 ms (100%)	1
MMAGENT[ENVIR-Virtual_3D]		22805 ms (100%)	1
MMAGENT[AGENT-cognitive]		22732 ms (100%)	1
MMAGENT[INTER-Rules_based]		22701 ms (100%)	1
MMAGENT[ENVIR-Competence]		22528 ms (100%)	1
MMAGENT[ORGAN-Group_organization]		22376 ms (100%)	1
MMAGENT[INTER-Dialog_based]		22149 ms (100%)	1
MMAGENT[ORGAN-Hierarchical]		21909 ms (100%)	1
OBSERVER		13449 ms (100%)	1
Thread-10		1859 ms (100%)	1
Thread-9		66.5 ms (100%)	1
Thread-8		58.5 ms (100%)	1
main		32.1 ms (100%)	1

Figure 8.5: Profiling Agents Performance

The agents performance is presented in the figure 8.5 where is registered the overall execution time of each agent. We can observe that each agent needs an approximate time of 23000 ms to perform their task. The Observer process requires almost 14000 ms.

Moreover the behavior of the agents is composed by several actions that we can observe within each agent in the figure 8.6. The two most required methods are the setup that is executed only one time but initializes each agent and the Negotiation actions that are peformed 22 times.

The setup method that is displayed in the figure 8.7 initializes the overall agent behavior. It takes the micro-arrays of the characterized meta-model and the AS-characterization. It also creates the local agent group and launches the communication services. Such a method instantiates the agent itself in order to make it operational. It also recovers the addresses of the other agents in order to communicate with them.

Analyzing the Negotiation actions we can observe in the figure 8.8 that the most employed methods are related to the decision making about the group creation. The setting the agent owner values and performing the certainty (probability of success) of adding a new member in the group.

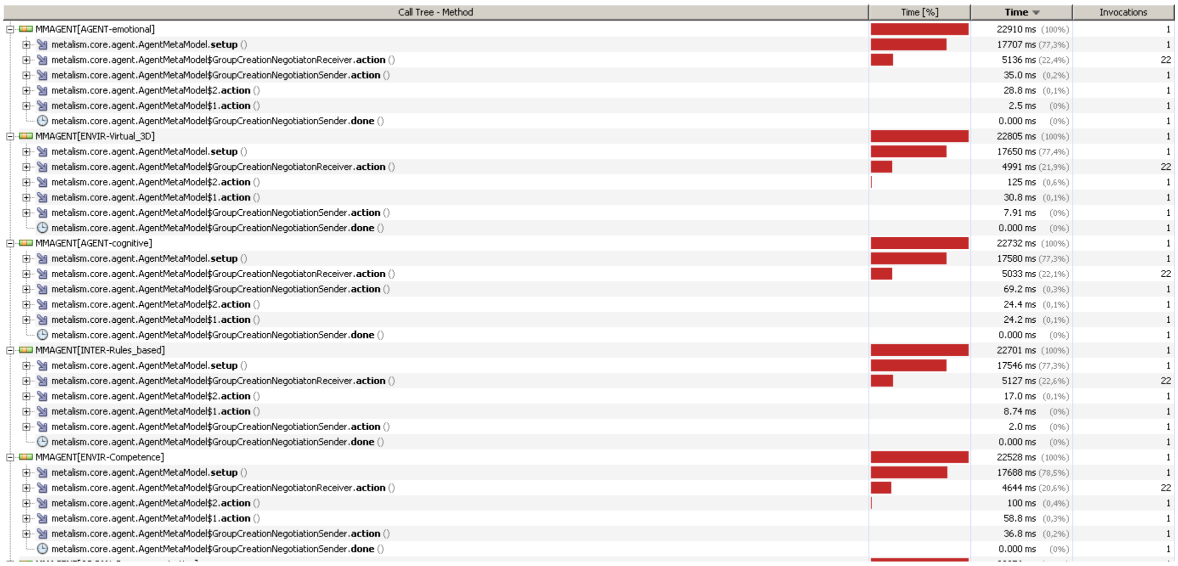


Figure 8.6: Profiling Agent Behavior performance within the tool execution

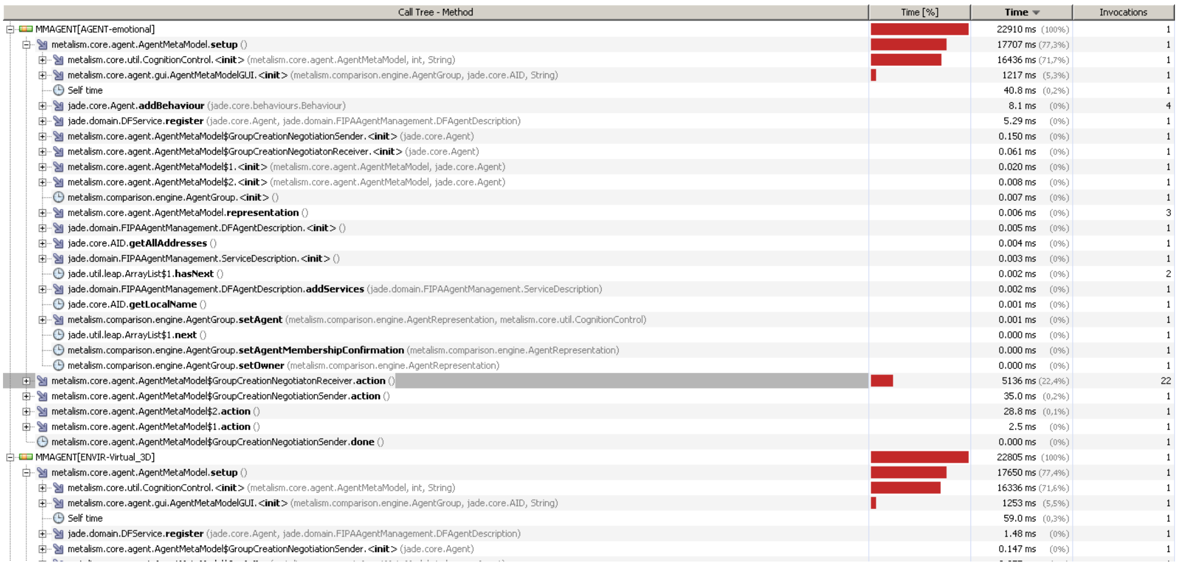
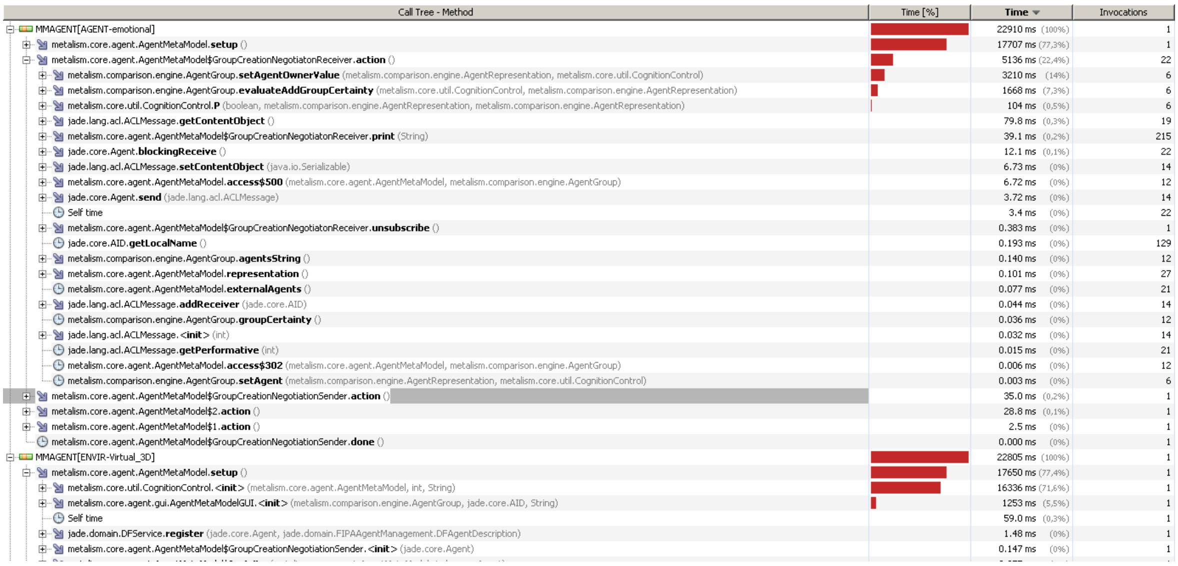


Figure 8.7: Profiling Making Decision Agent Actions

There is also the Probability of being solution for a pair of meta-models.

In conclusion the general time required to perform the tool is less than 1 minute. Nevertheless considering that the operations that require most of the execution time are the read/write operations (usually considered as basic operation) the complexity of the tool is efficient. Also we must consider that the induction algorithm (Probability method), as one of the most executed methods, has a polynomial complexity. Then the overall time required for the tool execution independently of the hardware is considered as efficient.



Call Tree - Method	Time [%]	Time	Invocations
MMAGENT[AGENT-emotional]		22910 ms (100%)	1
metalism.core.agent.AgentMetaModel.**setup** ()		17707 ms (77,3%)	1
metalism.core.agent.AgentMetaModel$GroupCreationNegotiatonReceiver.**action** ()		5136 ms (22,4%)	22
metalism.comparison.engine.AgentGroup.**setAgentOwnerValue** (metalism.comparison.engine.AgentRepresentation, metalism.core.util.CognitionControl)		3210 ms (14%)	6
metalism.comparison.engine.AgentGroup.**evaluateAddGroupCertainty** (metalism.core.util.CognitionControl, metalism.comparison.engine.AgentRepresentation)		1668 ms (7,3%)	6
metalism.core.util.CognitionControl.**P** (boolean, metalism.comparison.engine.AgentRepresentation, metalism.comparison.engine.AgentRepresentation)		104 ms (0,5%)	6
jade.lang.acl.ACLMessage.**getContentObject** ()		79.8 ms (0,3%)	19
metalism.core.agent.AgentMetaModel$GroupCreationNegotiatonReceiver.**print** (String)		39.1 ms (0,2%)	215
jade.core.Agent.**blockingReceive** ()		12.1 ms (0,1%)	22
jade.lang.acl.ACLMessage.**setContentObject** (java.io.Serializable)		6.73 ms (0%)	14
metalism.core.agent.AgentMetaModel.**access$500** (metalism.core.agent.AgentMetaModel, metalism.comparison.engine.AgentGroup)		6.72 ms (0%)	12
jade.core.Agent.**send** (jade.lang.acl.ACLMessage)		3.72 ms (0%)	14
Self time		3.4 ms (0%)	22
metalism.core.agent.AgentMetaModel$GroupCreationNegotiatonReceiver.**unsubscribe** ()		0.383 ms (0%)	1
jade.core.AID.**getLocalName** ()		0.193 ms (0%)	129
metalism.comparison.engine.AgentGroup.**agentsString** ()		0.140 ms (0%)	12
metalism.core.agent.AgentMetaModel.**representation** ()		0.101 ms (0%)	27
metalism.core.agent.AgentMetaModel.**externalAgents** ()		0.077 ms (0%)	21
jade.lang.acl.ACLMessage.**addReceiver** (jade.core.AID)		0.044 ms (0%)	14
metalism.comparison.engine.AgentGroup.**groupCertainty** ()		0.036 ms (0%)	12
jade.lang.acl.ACLMessage.**<init>** (int)		0.032 ms (0%)	14
jade.lang.acl.ACLMessage.**getPerformative** (int)		0.015 ms (0%)	21
metalism.core.agent.AgentMetaModel.**access$302** (metalism.core.agent.AgentMetaModel, metalism.comparison.engine.AgentGroup)		0.006 ms (0%)	12
metalism.comparison.engine.AgentGroup.**setAgent** (metalism.comparison.engine.AgentRepresentation, metalism.core.util.CognitionControl)		0.003 ms (0%)	6
metalism.core.agent.AgentMetaModel$GroupCreationNegotiationSender.**action** ()		35.0 ms (0,2%)	1
metalism.core.agent.AgentMetaModel$2.**action** ()		28.8 ms (0,1%)	1
metalism.core.agent.AgentMetaModel$1.**action** ()		2.5 ms (0%)	1
metalism.core.agent.AgentMetaModel$GroupCreationNegotiationSender.**done** ()		0.000 ms (0%)	1
MMAGENT[ENVIR-Virtual_3D]		22805 ms (100%)	1
metalism.core.agent.AgentMetaModel.**setup** ()		17650 ms (77,4%)	1
metalism.core.util.CognitionControl.**<init>** (metalism.core.agent.AgentMetaModel, int, String)		16336 ms (71,6%)	1
metalism.core.agent.gui.AgentMetaModelGUI.**<init>** (metalism.comparison.engine.AgentGroup, jade.core.AID, String)		1253 ms (5,5%)	1
Self time		59.0 ms (0,3%)	1
jade.domain.DFService.**register** (jade.core.Agent, jade.domain.FIPAAgentManagement.DFAgentDescription)		1.48 ms (0%)	1
metalism.core.agent.AgentMetaModel$GroupCreationNegotiationSender.**<init>** (jade.core.Agent)		0.147 ms (0%)	1

Figure 8.8: Profiling Agent Interaction Actions

8.7 Discussion

Comparing our results with a similar study described in (Barthés et al., 2002) where a comparison between seven different platforms is achieved using a similar auction situation, we can see that the type of agent used in each platform has features of a BDI agent (Wooldridge, 2000). This type of agent is linked to the type of cognitive agent has been chosen in our approach. However, our case study uses a limited set of features and not all of them are on all the platforms used in (Barthés et al., 2002) and vice versa, but we can see that our solutions have proved to be nearest to the most appropriate. So, this shows that the present approach based in a self-organized MAS is promising and we will continue working to improve and increase this approach.

A MAS study described in (Barthés et al., 2002) compares the solutions for an auction problem deployed in seven different MAS platforms. The agent type employed in each platform is a BDI agent (Wooldridge, 2000). Such agent type is linked to a cognitive agent type. In the work of (Guttman and Maes, 1998) is discussed a study about the environment and interaction protocols in agent auction situations, where competitive and group negotiation interaction is performed for some kind of auctions. About the organization in (Zhang et al., 2009) we find a study about the virtual organizations management where there are different categories of agent roles in an auction kind virtual organization. Thus our platform has chosen the group selected from the cognitive agent perspective. It means that the first decision making step has been to choose the most suitable meta-model agent type, immediately, select the rest of meta-model types considering the compatibility with the auction problem and with the meta-model agent type.

We can see that the optimal group showed in the figure 8.9 is based on (Barthés et al., 2002) and the group found in the test case is the winning group A. The rest are some of the groups created but with less probability of success.

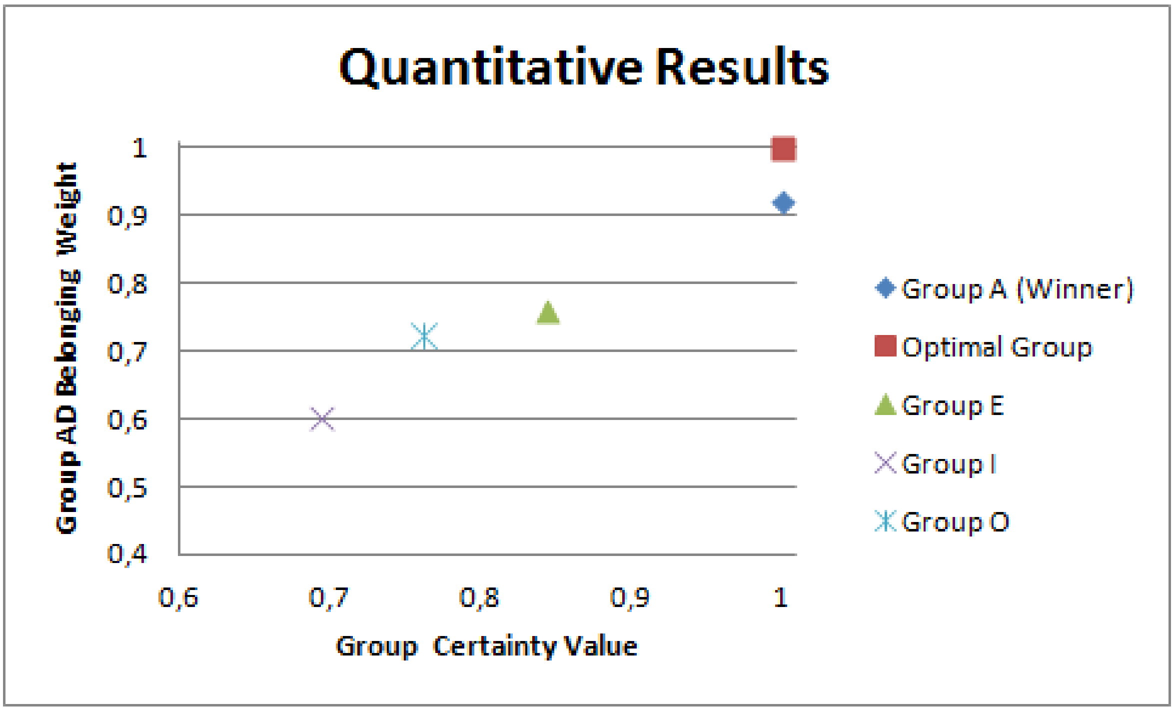


Figure 8.9: Quantitative Results obtained with the tool execution

Nevertheless we employ in our prototype a reduced meta-model features set and not all of them are on the components used in (Barthés et al., 2002; Guttman and Maes, 1998; Zhang et al., 2009). But we can perceive that the meta-model group found as most suitable solution is close to the right components.

Chapter 9

Conclusion and Future Work

9.1 Conclusions

The present thesis is the first step within a full path to develop a functional preliminary phase before to start a MAS-based software solution. Such a full path is in the fig. 6.2. Then we can see that the user input is the application specification, therefore it is characterized in order to be employed by the MAS-based comparison engine. Such a matching engine is the main contribution of this thesis work. We have also consider the second input of such an engine is the solution components ontology that is composed by MAS meta-models characterization under the vowels approach.

Therefore, we have proposed to create a matching engine that needs two characterized data sources. The process to characterize such data sources is different according to the data nature. Therefore our approach proposes to characterize the following data sources in the next manner:

- The problem description. This characterization consists in identifying a problem at least partially using a minimal knowledge base of known problem characteristics. Therefore identify the domain of the problem considering the ensemble of characteristics found. Such a characterization problem is considered as a satellite problem that is not the main focus of this thesis, however, we have solved such problem employing a minimal approach described in the section 5.3.1. However such approach could be improved and extended as described in the section 9.2 or changed completely. The key feature is that we need to treat with a high grade of uncertainty when we identify at least partially a problem. Therefore t in order to identify their domain and thus known which specialized entities could be employed as part of a solution. Finally we have
- The specialized entities. In contrast with the problem description the specialized entities has not a high uncertainty grade. They are entities that are usually well-known and stored, in our case using OWL for meta-models or UML, therefore the characterization could be performed employing a semantic based analysis. In our specific case we have defined the meta-models as a set of features, where each feature has relations with many domains represented by efficacy

values.

The experience values are the most sensible feature of our approach because is very important to have at least a minimum of experience acquired within such an experience data. There are many possible ways to acquire such experience, between them:

- On-line learning. It is to acquire the experience from the outcome obtained each time that the system evaluates a problem and proposes a solution with success or not. This is performed updating the experience values for each problem characteristic and meta-model feature selected.
- Surveys. Employing the surveys we can gather the experience from human experts in different matter. Such surveys can be generated automatically considering the data within the existing experience knowledge bases. Therefore the human experts must answer boolean questions on a match cross board about the problem characteristics and meta-model features relationship and about two sets of meta-model features of different type compatibility within a domain. Such data will be stored and employed as experience for the meta-model candidate selection and for the making decision to build groups within the agents.
- Automated analysis. This analysis must be done using the current solutions and methodologies as data to be analyzed in order to abstract from them the required experience about the relationships of problem characteristics - meta-models features efficacy and the meta-models features compatibility.

We have worked in the central part of such project (See fig. 6.2. Such part is the matching engine where the matching process is performed.

Therefore resuming in this thesis work we have created a process to match problems characteristics with solutions features in order to build solutions. Nevertheless to arrive a such process we have reached a diverse background in many areas:

- Thermodynamics. We have taken the conceptual ideas from the second law of thermodynamics and the Maxwell's demon paradox(Kelvin, 1879; Kelvin, 1892) in order to achieve an abstract concept about our problem and locate it within the information technology domain.
- Society. The MAS organization has been inspired in the society ideas from (Minsky, 1986) about a society of specialized and managers agents that work ensemble in order to create an intelligent system.
- Uncertainty. We have found that our approach need to treat with a considerable amount of uncertainty when it receives the problem. Therefore in order to treat with such uncertainty we have found several interesting ways to do that with robots and automatic systems in (Bessière et al., 2008).

- Statistics. We have taken the agent inference algorithm from the succession's law of Pierre Laplace (Laplace, 1812) in order to make decisions considering a small amount of historic data. This is our specific case because we have no a big experience data but we have also defined a way to do that employing meta-analysis methods (Leandro, 2005), but in such case we need a biggest amount of experience data.

- Artificial Intelligence. We have employed Bayesian cognition (Bessière et al., 2008) in order to treat with uncertainty and considering the statistical nature of meta-analysis.

- Software Engineering. Considering the MAS decomposition and component re usability issues we have analyzed some of the existing options of process fragments, covered by the work of (Mariachiara et al., 2010) and (Massimo et al., 2006) and MAS meta-modelization covered into (Ferber and Gutknecht, 1998b), (Z.Guessoum and Jarraya, 2005), (Bernon et al., 2004), (Giret and Botti, 2004) and (Ferber and Gutknecht, 1998c) to reach a solution. Therefore we have considered the vowels approach (Demazeau, 1995) and the Model-Driven Engineering (MDE) approach (Gasevic et al., 2009b) to conceive the idea of a repository of MAS meta-models created using such approaches. However in our work (matching engine context) we have employed only a characterization of such MAS meta-models.

- MAS. We have employed a MAS within the matching engine with the perspective of recreate the human expert designer and meta-analysis roles. Usually is a human designer expert that must design a MAS from the development view of a type of MAS component, also there is a human meta-analysis expert that performs the meta-analysis and interpret the results to make decisions. Also usually there is a group of human experts that works together to build such solutions. Thus, we have employed all the previous items of this list to build the MAS within the matching engine.

However the present approach is composed by many contributions that has been created through the research that we list in the following list:

- We propose a manager agent society that employs meta-models of specialized components. It has been inspired in the work proposed by Marvin Minsky but we propose a unique kind of manager agents that work together to build individual solution groups considering the development views. Each development view considers the perspective of each kind of specialized component to build the solution similar as proposed in (Jürgen, 2001). In other words each agent represents a specialized component and it acts to build a solution that is the more convenient for such a specialized component.

- We propose a meta-analysis performed within each agent making decision process. The meta-analysis nature allows to create agents that could infer the success probability of their actions using statistical data and then make decisions. It also permits to accept uncertainty in the agents perception in order to make decisions in despite of the unclear conditions.

- We have proposed a knowledge model to manage the problem characteristics, meta-model features and domain in order to be employed within the matching engine.
- We propose an approach as basis of a future work about a global knowledge base enriched by many designers and MAS experts around the world seeking to improve the approach results. This is something similar as pubmed is for the medical domain.
- The whole matching engine can be generalized to be applied in many other kinds of related problems, for example choosing test cases for software testing, choosing robots with different skills to solve a group task, OSGi package composition to create dynamically new solutions, etc.
- We have also contributed with the prototype tool that implements the matching engine. [1].

9.2 Future Work

Our approach is a new alternative for the developers to encourage them in the use of MAS based solutions. This approach is positioned as a preliminary phase of software engineering where the system designer can evaluate the MAS as a possible way to follow. The approach described here is the first part of a complex solution that is still under development; nevertheless it considers important aspects, such as re usability of existing MAS models and update capability of the domain-problem ontology and meta-model ontology. The micro-array profits these upgradeable aspects. The solution emerges from matching performed in self-organized MAS, where the meta-models characterization acts as an agent with certain behavior oriented to looking for another type of meta-model agents. Once such agents are found, the matching is performed and relations between them are created. The last part consists in meta-analysis of the set of possible solutions, where the certitude criterion is applied to find the most accurate solution. The final result is an MAS meta-model set.

Future work for this approach is rough because it includes an entire project. The future work includes the complete meta-analysis already done in this thesis with a low level that will be based on proposals existing solutions. This will provide feedback to the entire system and allow the future creation of a knowledge base that will contain the link orderly from problem (problem features) to solution (solution components). Which will make for a meta-analysis process at a high level for trends of hosted solutions in that knowledge base and identify components and possible solution.
On the other hand it is necessary to address the satellite phases more deeply to be made in this thesis. The characterizations of the problem and meta-models must be done as two separate approaches. For the characteristics of problem we need to create a method to automatically acquire these characteristics and be able to stay within a knowledge base linking them to the domains of

[1] Currently in closed alpha release and available soon at the website https://sourceforge.net/projects/metalism/

solution, however we have created a minimal approach described in the Appendix B. For the meta-models we can create an automatic method to abstract meta-model solutions created with existing methodologies, however, it could be by hand.

The advantages of this thesis approach is that it can deal with uncertainty, adapting dynamically and can be distributed. Also it can be generalized and applied to solve other kinds of problems beyond the MAS software engineering.

The disadvantage is that its effectiveness depends heavily on experience to the decision-making. With a low level of experience is difficult to reach a reliable proposed solution. That's why the acquisition of experience is connected with the feedback system, which has been proposed as future work to automatically feed the system with positive or negative results obtained. Thereby increasing the efficiency and trust ability on the outcome.

Finally this work, as mentioned, can be generalized and employed in many other types of problem related to the comparison and distributed decision making. By their nature based on the comparison engine agents can be distributed. This would allow its implementation in different physical or virtual entities. The construction of solutions could be from physical components or software, or static nature of actors follow a script dynamically created to act and solve problems. The latter opens up endless possibilities to be used and improved in many areas is thus a useful and innovative approach.

Part V

Appendices

Appendix A

XML Knowledge base Files

Problem - Meta-metamodels Match compatibility

The following listing displays an extract from the XML-based knowledge base that enables to match problem characteristics with meta-models features.

Listing 9.1: Problem Characteristics and Meta-Models Features Compatibiliy Knowledge base

```
<?xml version="1.0" encoding="utf-8"?>
<match-registry application-specification="Auction␣Problem" meta-
   models-KB="&#xA;&#x9;Meta-models␣repository&#xA;&#x9;&#xA;&#x9
   ;&#xA;" application-specification-number="18" meta-models-KB-
   number="64" domains-number="3">
<domains>
<domain id="1" name="World␣Simulation">This domain covers the
   simulation area. Generally virtual worlds, virtual drama,
   physical simulations,etc</domain>
<domain id="2" name="General␣Bussiness">This domain comprises the
    bussiness field with the sell-buy and transaction details and
    features.</domain>
<domain id="3" name="Sales␣context">The Sales Context domain
   covers the sell and buy features, including auctions, offer
   and demand laws, special offers, etc.</domain>
</domains>
<matrix-registry>
...
<registry domain-id="3" name="Sales␣context" true="35" false="29"
   >
<match-value id="1" as-characteristic-id="1" mm-characteristic-id
   ="1" pos-as-characteristic="0" pos-mm-feature="0">
<record n="false" value="0">
```

```
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="0" />
</record>
<record n="true" value="0">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="0" />
</record>
</match-value>
...
<match-value id="129" as-characteristic-id="3" mm-characteristic-
   id="1" pos-as-characteristic="2" pos-mm-feature="0">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="130" as-characteristic-id="3" mm-characteristic-
   id="2" pos-as-characteristic="2" pos-mm-feature="1">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
```

```
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="131" as-characteristic-id="3" mm-characteristic-
   id="3" pos-as-characteristic="2" pos-mm-feature="2">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="132" as-characteristic-id="3" mm-characteristic-
   id="4" pos-as-characteristic="2" pos-mm-feature="3">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="133" as-characteristic-id="3" mm-characteristic-
   id="5" pos-as-characteristic="2" pos-mm-feature="4">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
```

```
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="134" as-characteristic-id="3" mm-characteristic-
   id="6" pos-as-characteristic="2" pos-mm-feature="5">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="135" as-characteristic-id="3" mm-characteristic-
   id="7" pos-as-characteristic="2" pos-mm-feature="6">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
```

```
</match-value>
<match-value id="136" as-characteristic-id="3" mm-characteristic-
   id="8" pos-as-characteristic="2" pos-mm-feature="7">
<record n="false" value="11">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="11" />
</record>
<record n="true" value="15">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="15" />
</record>
</match-value>
<match-value id="137" as-characteristic-id="3" mm-characteristic-
   id="9" pos-as-characteristic="2" pos-mm-feature="8">
<record n="false" value="18">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="18" />
</record>
<record n="true" value="20">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="20" />
</record>
</match-value>
<match-value id="138" as-characteristic-id="3" mm-characteristic-
   id="10" pos-as-characteristic="2" pos-mm-feature="9">
<record n="false" value="18">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="18" />
        <match theta="true" mu="true" value="0" />
</record>
```

```
<record n="true" value="20">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="20" />
        <match theta="true" mu="true" value="0" />
</record>
</match-value>
<match-value id="139" as-characteristic-id="3" mm-characteristic-
   id="11" pos-as-characteristic="2" pos-mm-feature="10">
<record n="false" value="18">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="18" />
</record>
<record n="true" value="20">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="20" />
</record>
</match-value>
<match-value id="140" as-characteristic-id="3" mm-characteristic-
   id="12" pos-as-characteristic="2" pos-mm-feature="11">
<record n="false" value="18">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="18" />
        <match theta="true" mu="true" value="0" />
</record>
<record n="true" value="20">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="20" />
        <match theta="true" mu="true" value="0" />
</record>
</match-value>
<match-value id="141" as-characteristic-id="3" mm-characteristic-
   id="13" pos-as-characteristic="2" pos-mm-feature="12">
```

```
<record n="false" value="18">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="18" />
</record>
<record n="true" value="20">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="0" />
        <match theta="true" mu="true" value="20" />
</record>
</match-value>
<match-value id="142" as-characteristic-id="3" mm-characteristic-
   id="14" pos-as-characteristic="2" pos-mm-feature="13">
<record n="false" value="18">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="18" />
        <match theta="true" mu="true" value="0" />
</record>
<record n="true" value="20">
        <match theta="false" mu="false" value="0" />
        <match theta="false" mu="true" value="0" />
        <match theta="true" mu="false" value="20" />
        <match theta="true" mu="true" value="0" />
</record>
</match-value>
...
                </registry>
        </matrix-registry>
</match-registry>
```

Unification Registry between Meta-models

The next listing discloses a part of the Unification Registry XML-based knowledge base.

Listing 9.2: Meta-Models Compatibility Knowledge Base

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<unification-registry meta-models-KB="&#xA;&#x9;Meta-models␣
   repository&#xA;&#x9;&#xA;&#x9;&#xA;" meta-models-KB-number="64
   " domains-number="3">
<hyper-unification-matrix-registry elements="3">
<unification-matrix-registry domain-id="3" name="Sales␣context">
<matrix true="10" false="0" tau-i="1" type-i="0" string-type-i="
   agent" tau-j="3" type-j="1" string-type-j="environment">
<unification-value id="1" mm-features-i-id="1" mm-features-j-id="
   17" pos-mm-features-i="0" pos-mm-features-j="0">
<record n="false" value="8">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="8" />
</record>
<record n="true" value="32">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="32" />
</record>
</unification-value>
...
</matrix>
<matrix true="0" false="0" tau-i="1" type-i="0" string-type-i="
   agent" tau-j="4" type-j="1" string-type-j="environment">
<unification-value id="1" mm-features-i-id="1" mm-features-j-id="
   25" pos-mm-features-i="0" pos-mm-features-j="0">
<record n="false" value="15">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="15" />
</record>
<record n="true" value="15">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="15" />
```

```
</record>
</unification-value>
...
</matrix>
<matrix true="0" false="0" tau-i="1" type-i="0" string-type-i="
   agent" tau-j="5" type-j="2" string-type-j="interaction">
<unification-value id="1" mm-features-i-id="1" mm-features-j-id="
   33" pos-mm-features-i="0" pos-mm-features-j="0">
<record n="false" value="18">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="18" />
</record>
<record n="true" value="12">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="12" />
</record>
</unification-value>
...
</matrix>
<matrix true="10" false="0" tau-i="1" type-i="0" string-type-i="
   agent" tau-j="6" type-j="2" string-type-j="interaction">
<unification-value id="1" mm-features-i-id="1" mm-features-j-id="
   41" pos-mm-features-i="0" pos-mm-features-j="0">
<record n="false" value="5">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="5" />
</record>
<record n="true" value="35">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="35" />
</record>
```

```
</unification-value>
...
<matrix true="0" false="0" tau-i="1" type-i="0" string-type-i="
   agent" tau-j="7" type-j="3" string-type-j="organization">
<unification-value id="1" mm-features-i-id="1" mm-features-j-id="
   49" pos-mm-features-i="0" pos-mm-features-j="0">
<record n="false" value="18">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="18" />
</record>
<record n="true" value="14">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="14" />
</record>
</unification-value>
...
</matrix>
<matrix true="10" false="0" tau-i="1" type-i="0" string-type-i="
   agent" tau-j="8" type-j="3" string-type-j="organization">
<unification-value id="1" mm-features-i-id="1" mm-features-j-id="
   57" pos-mm-features-i="0" pos-mm-features-j="0">
<record n="false" value="5">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="5" />
</record>
<record n="true" value="33">
        <unification mu-i="false" mu-j="false" value="0" />
        <unification mu-i="false" mu-j="true" value="0" />
        <unification mu-i="true" mu-j="false" value="0" />
        <unification mu-i="true" mu-j="true" value="33" />
</record>
</unification-value>
...
```

```
</matrix>
...
</unification-matrix-registry>
</hyper-unification-matrix-registry>
</unification-registry>
```

OWL Ontology File

The code showed lines below shows a fragment from the ontology employed in this thesis. We can see the problem characteristics defined and the meta-models types and features.

Listing 9.3: OWL Ontology file

```
<?xml version="1.0"?>


<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<Ontology xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.semanticweb.org/ontologies/2011/6/AS-
        problem-domain.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     ontologyIRI="http://www.semanticweb.org/ontologies/2011/6/AS
        -problem-domain.owl"
     versionIRI="http://www.semanticweb.org/ontologies/2011/6/AS-
        problem-domain.owl">
    <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
    <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-
       syntax-ns#"/>
```

```
<Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema
   #"/>
<Declaration>
    <Class IRI="#
       ActorAcquireExperienceManagingPosturesCharacteristic"/
       >
</Declaration>
<Declaration>
    <Class IRI="#
       ActorFacialExpressionBodyPostureEmotionCharacteristic"
       />
</Declaration>
<Declaration>
    <Class IRI="#ActorFollowDefinedRulesCharacteristic"/>
</Declaration>
    ...
<Declaration>
    <Class IRI="#MetaModel"/>
</Declaration>
<Declaration>
    <Class IRI="#MetaModelFeature"/>
</Declaration>
<Declaration>
    <Class IRI="#MetaModelType"/>
</Declaration>
<Declaration>
    <Class IRI="#MinimalSetOfWordsMMFeature"/>
</Declaration>
    ...
<Declaration>
    <ObjectProperty IRI="#hasDomain"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#hasFeature"/>
</Declaration>
<Declaration>
    <ObjectProperty IRI="#hasType"/>
</Declaration>
<Declaration>
```

```
        <ObjectProperty IRI="#hasValue"/>
    </Declaration>
    <Declaration>
        <ObjectProperty IRI="#isDomainOf"/>
    </Declaration>
    <Declaration>
        <ObjectProperty IRI="#isFeatureOf"/>
    </Declaration>
    <Declaration>
        <ObjectProperty IRI="#isTypeOf"/>
    </Declaration>
    <SubClassOf>
        <Class IRI="#
           ActorAcquireExperienceManagingPosturesCharacteristic"/
           >
        <Class IRI="#ProblemCharacteristic"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#
           ActorAcquireExperienceManagingPosturesCharacteristic"/
           >
        <ObjectIntersectionOf>
            <ObjectSomeValuesFrom>
                <ObjectProperty IRI="#hasDomain"/>
                <Class IRI="#NegotiationBasedDomain"/>
            </ObjectSomeValuesFrom>
            <ObjectExactCardinality cardinality="5">
                <ObjectProperty IRI="#hasValue"/>
            </ObjectExactCardinality>
        </ObjectIntersectionOf>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#
           ActorFacialExpressionBodyPostureEmotionCharacteristic"
           />
        <Class IRI="#ProblemCharacteristic"/>
    </SubClassOf>
    <SubClassOf>
```

```
        <Class IRI="#
           ActorFacialExpressionBodyPostureEmotionCharacteristic"
           />
        <ObjectIntersectionOf>
            <ObjectSomeValuesFrom>
                <ObjectProperty IRI="#hasDomain"/>
                <Class IRI="#NegotiationBasedDomain"/>
            </ObjectSomeValuesFrom>
            <ObjectExactCardinality cardinality="5">
                <ObjectProperty IRI="#hasValue"/>
            </ObjectExactCardinality>
        </ObjectIntersectionOf>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ActorFollowDefinedRulesCharacteristic"/>
        <Class IRI="#ProblemCharacteristic"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ActorFollowDefinedRulesCharacteristic"/>
        <ObjectIntersectionOf>
            <ObjectSomeValuesFrom>
                <ObjectProperty IRI="#hasDomain"/>
                <Class IRI="#NegotiationBasedDomain"/>
            </ObjectSomeValuesFrom>
            <ObjectExactCardinality cardinality="5">
                <ObjectProperty IRI="#hasValue"/>
            </ObjectExactCardinality>
        </ObjectIntersectionOf>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#AgentMetaModel"/>
        <Class IRI="#MetaModelType"/>
    </SubClassOf>
        ...
    <SubClassOf>
        <Class IRI="#Virtual3DEnvironmentCMM"/>
        <ObjectAllValuesFrom>
            <ObjectProperty IRI="#hasType"/>
            <Class IRI="#EnvironmentMetaModel"/>
```

```
        </ObjectAllValuesFrom>
    </SubClassOf>
    <DisjointClasses>
        <Class IRI="#
           ActorAcquireExperienceManagingPosturesCharacteristic"/
           >
        <Class IRI="#
           ActorFacialExpressionBodyPostureEmotionCharacteristic"
           />
        <Class IRI="#ActorFollowDefinedRulesCharacteristic"/>
                ...
        <Class IRI="#PriceNegotiationProcessCharacteristic"/>
        <Class IRI="#SharingCostsCharacteristic"/>
        <Class IRI="#SimilarPhysicalMovementCharacteristic"/>
        <Class IRI="#TaxReductionHintsCharacteristic"/>
    </DisjointClasses>
    <DisjointClasses>
        <Class IRI="#AgentMetaModel"/>
        <Class IRI="#EnvironmentMetaModel"/>
        <Class IRI="#InteractionMetaModel"/>
        <Class IRI="#OrganizationMetaModel"/>
    </DisjointClasses>
    <DisjointClasses>
        <Class IRI="#AnswerResponseSequencesMMFeature"/>
        <Class IRI="#BasicRuleLanguageMMFeature"/>
        <Class IRI="#BroadcastCommunicationMMFeature"/>
        <Class IRI="#CommunicationPermissionsMMFeature"/>
                ...
        <Class IRI="#StrategiccompetenceMMFeature"/>
        <Class IRI="#TacticActionsAndDecisionsMMFeature"/>
        <Class IRI="#ThirdAgentsRepresentationMMFeature"/>
        <Class IRI="#UsesKnowledgeAboutOrganizationMMFeature"/>
    </DisjointClasses>
    <DisjointClasses>
        <Class IRI="#CognitiveAgentCMM"/>
        <Class IRI="#CompetenceEnvironmentCMM"/>
        <Class IRI="#DialogBasedInteractionCMM"/>
        <Class IRI="#EmotionalAgentCMM"/>
        <Class IRI="#GroupOrganizationCMM"/>
```

```
        <Class IRI="#HierarchicalOrganizationCMM"/>
        <Class IRI="#RuleBasedInteractionCMM"/>
        <Class IRI="#Virtual3DEnvironmentCMM"/>
    </DisjointClasses>
    <DisjointClasses>
        <Class IRI="#NegotiationBasedDomain"/>
        <Class IRI="#PublicSalesContextDomain"/>
        <Class IRI="#SimulationDomain"/>
    </DisjointClasses>
    <InverseObjectProperties>
        <ObjectProperty IRI="#isDomainOf"/>
        <ObjectProperty IRI="#hasDomain"/>
    </InverseObjectProperties>
    <InverseObjectProperties>
        <ObjectProperty IRI="#isFeatureOf"/>
        <ObjectProperty IRI="#hasFeature"/>
    </InverseObjectProperties>
    <InverseObjectProperties>
        <ObjectProperty IRI="#hasType"/>
        <ObjectProperty IRI="#isTypeOf"/>
    </InverseObjectProperties>
    <FunctionalObjectProperty>
        <ObjectProperty IRI="#hasValue"/>
    </FunctionalObjectProperty>
    <TransitiveObjectProperty>
        <ObjectProperty IRI="#hasDomain"/>
    </TransitiveObjectProperty>
    <TransitiveObjectProperty>
        <ObjectProperty IRI="#hasFeature"/>
    </TransitiveObjectProperty>
    <TransitiveObjectProperty>
        <ObjectProperty IRI="#hasType"/>
    </TransitiveObjectProperty>
    <TransitiveObjectProperty>
        <ObjectProperty IRI="#isDomainOf"/>
    </TransitiveObjectProperty>
    <TransitiveObjectProperty>
        <ObjectProperty IRI="#isFeatureOf"/>
    </TransitiveObjectProperty>
```

```
    <TransitiveObjectProperty>
        <ObjectProperty IRI="#isTypeOf"/>
    </TransitiveObjectProperty>
</Ontology>
```

Application Specification XML File Example

An example of the XML standard employed to store AS-characterization is as follows:

- The header contains the title, author and contact information:

```
<Application-Specification-File>
<as-characterization id="0" title="Auction Problem">
<owner author="Luis " email="lrazo1@gmail.com" institution="INPG"/>
```

- Therefore there is a text that describes the application specification

```
<application-specification>
A system that represents a virtual
auction salon.
There two main kind of actors:
- Customers with two roles: sellers and bidders.
- And the auctioneer who performs the auction
following the rules specified.
Both customers actors, buyers and bidders, can
create their representatives customizing a
behavior as they want, a virtual entity that
represents the customer in the virtual auction
salon. These virtual entity has a customizable
behavior it means that the real customer can add
the following features:
- A maximum budget for a list or category of
articles.
- Also they could accept rules to make or not bids.
In the other side the virtual salon will be managed
by an automated virtual entity that will create
auctions taking proposed goods to sell from
the customers that want to sell in the auction.
So we will accept goods with a initial value and
a desired selling value so we will add a 20% of
```

```
management cost to the final selling value.
Every auction could be customizable in the way
that the selling customer could chose between
four main types of auction rules:
- English auction,
- Dutch auction,
- Sealed first-price auction,
- Vickrey auction.
</application-specification>
```

- Then the domains descriptions are stored:

```
<characterization-map>
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
```

- Moreover the list of known problem characteristics are listed:

```
<characteristics>
<characteristic id="1" name="3D Virtual actors">An environment with 3D
virtual actors</characteristic>
<characteristic id="2" name="Opponent movement predictions">Analyze the
opponent and predict his movements or decisions</characteristic>
<characteristic id="3" name="Auction actor roles">Auction actors are
customers with buyer and seller roles</characteristic>
<characteristic id="4" name="Auction process control">Auctioneer actor
controls the auction process</characteristic>
<characteristic id="5" name="English auction protocol">English auction
protocol</characteristic>
<characteristic id="6" name="Efficient taxes payment">Find the best
hints to reduce the taxes payment</characteristic>
<characteristic id="7" name="Debt contract constraint">if the debtor
```

```
fails to pay the debt according to the contract shall be entitled as
lien</characteristic>
<characteristic id="8" name="Optimize the income">Maximize the income
or gain</characteristic>
<characteristic id="9" name="Cost optimization">Minimize the cost or
price</characteristic>
<characteristic id="10" name="Real Physical movements">Movements should
be considered as similar to real physical</characteristic>
<characteristic id="11" name="Auction bid restriction">Number of bids
limited</characteristic>
<characteristic id="12" name="On-line virtual room">On-line virtual room
</characteristic>
<characteristic id="13" name="Price negotiation process">Price
negotiation process</characteristic>
<characteristic id="14" name="Cost sharing">Propose to pay costs and
sharing equally</characteristic>
<characteristic id="15" name="Virtual drama actor’s emotional features">
The actors express their emotions with facial expressions and body
postures</characteristic>
<characteristic id="16" name="Virtual drama rules feature">the actors
must follow the defined rules</characteristic>
<characteristic id="17" name="Virtual actor position and movement
self-knowledge and learning">The actors will learn to manage their
positions using their experience</characteristic>
<characteristic id="18" name="Investment and foreign exchange">The
investment considers the foreign exchange rates</characteristic>
</characteristics>
```


- Finally the tag the list of efficacy values that relates problem characteristics with each domain are defined:


```
<characterization>
<characterization id="1" domain="1" characteristic="1" value="0.985"/>
<characterization id="2" domain="2" characteristic="1" value="0.561"/>
<characterization id="3" domain="3" characteristic="1" value="0.655"/>
<characterization id="4" domain="1" characteristic="2" value="0.69"/>
<characterization id="5" domain="2" characteristic="2" value="0.715"/>
<characterization id="6" domain="3" characteristic="2" value="0.598"/>
<characterization id="7" domain="1" characteristic="3" value="0.532"/>
```

```
<characterization id="8" domain="2" characteristic="3" value="0.611"/>
<characterization id="9" domain="3" characteristic="3" value="0.971"/>
<characterization id="10" domain="1" characteristic="4" value="0.572"/>
<characterization id="11" domain="2" characteristic="4" value="0.702"/>
<characterization id="12" domain="3" characteristic="4" value="0.656"/>
<characterization id="13" domain="1" characteristic="5" value="0.41"/>
<characterization id="14" domain="2" characteristic="5" value="0.712"/>
<characterization id="15" domain="3" characteristic="5" value="0.795"/>
<characterization id="16" domain="1" characteristic="6" value="0.59"/>
<characterization id="17" domain="2" characteristic="6" value="0.699"/>
<characterization id="18" domain="3" characteristic="6" value="0.68"/>
<characterization id="19" domain="1" characteristic="7" value="0.189"/>
<characterization id="20" domain="2" characteristic="7" value="0.908"/>
<characterization id="21" domain="3" characteristic="7" value="0.759"/>
<characterization id="22" domain="1" characteristic="8" value="0.454"/>
<characterization id="23" domain="2" characteristic="8" value="0.785"/>
<characterization id="24" domain="3" characteristic="8" value="0.875"/>
<characterization id="25" domain="1" characteristic="9" value="0.433"/>
<characterization id="26" domain="2" characteristic="9" value="0.782"/>
<characterization id="27" domain="3" characteristic="9" value="0.915"/>
<characterization id="28" domain="1" characteristic="10" value="0.29"/>
<characterization id="29" domain="2" characteristic="10" value="0.89"/>
<characterization id="30" domain="3" characteristic="10" value="0.692"/>
<characterization id="31" domain="1" characteristic="11" value="0.5"/>
<characterization id="32" domain="2" characteristic="11" value="0.5"/>
<characterization id="33" domain="3" characteristic="11" value="0.5"/>
<characterization id="34" domain="1" characteristic="12" value="0.893"/>
<characterization id="35" domain="2" characteristic="12" value="0.48"/>
<characterization id="36" domain="3" characteristic="12" value="0.695"/>
<characterization id="37" domain="1" characteristic="13" value="0.119"/>
<characterization id="38" domain="2" characteristic="13" value="0.691"/>
<characterization id="39" domain="3" characteristic="13" value="0.898"/>
<characterization id="40" domain="1" characteristic="14" value="0.201"/>
<characterization id="41" domain="2" characteristic="14" value="0.854"/>
<characterization id="42" domain="3" characteristic="14" value="0.876"/>
<characterization id="43" domain="1" characteristic="15" value="0.944"/>
<characterization id="44" domain="2" characteristic="15" value="0.302"/>
<characterization id="45" domain="3" characteristic="15" value="0.389"/>
<characterization id="46" domain="1" characteristic="16" value="0.891"/>
```

```
<characterization id="47" domain="2" characteristic="16" value="0.595"/>
<characterization id="48" domain="3" characteristic="16" value="0.689"/>
<characterization id="49" domain="1" characteristic="17" value="0.864"/>
<characterization id="50" domain="2" characteristic="17" value="0.523"/>
<characterization id="51" domain="3" characteristic="17" value="0.37"/>
<characterization id="52" domain="1" characteristic="18" value="0.39"/>
<characterization id="53" domain="2" characteristic="18" value="0.823"/>
<characterization id="54" domain="3" characteristic="18" value="0.705"/>
</characterization>
</characterization-map>
</as-characterization>
</Application-Specification-File>
```

Meta-model characterization XML File Example

Also for the MM-characterization we have the next XML file example:

- The file header contains the name and the defined domains. We have considered to use the same domains employed in both XML files in order to keep a link. However is possible to create different domains creating a set of rules to create links between different domains. In this thesis we do not focus on this, therefore, such improvements are considered future work.

```
<?xml version="1.0" encoding="UTF-8"?>
<MM-Characterization-KB>
Meta-models repository
<domains>
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
```

- The next tag contains the list of defined meta-models, each meta-model has a name, a type, a list of domains, a list of their features and finally a list of efficacy values that relates each feature within each domain:

```
<meta-models>
<meta-model id="1" name="cognitive" type="0">
<domains>
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the businesses
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain></domains>
<characteristics>
<characteristic id="1" name="deliberative cognition">deliberative
cognition
</characteristic>
<characteristic id="2"
name="knowledge representation about itself and others">knowledge
representation about itself and others
</characteristic>
<characteristic id="3" name="strategic competence">strategic
competence
</characteristic>
<characteristic id="4" name="independent performance">independent
performance
</characteristic>
<characteristic id="5" name="cooperative behavior">cooperative
behavior
</characteristic>
<characteristic id="6"
name="uses knowledge about organization">uses knowledge about
organization</characteristic>
<characteristic id="7" name="social listening to others">social
  listening to others</characteristic>
<characteristic id="8" name="tactic actions and decisions">tactic
  actions and decisions</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="1"
```

```
value="0.875" />
<characterization id="2" domain="2" characteristic="1"
value="0.905" />
<characterization id="3" domain="3" characteristic="1"
value="0.855" />
<characterization id="4" domain="1" characteristic="2"
value="0.725" />
<characterization id="5" domain="2" characteristic="2"
value="0.85" />
<characterization id="6" domain="3" characteristic="2"
value="0.9" />
<characterization id="7" domain="1" characteristic="3"
value="0.8" />
<characterization id="8" domain="2" characteristic="3"
value="0.775" />
<characterization id="9" domain="3" characteristic="3"
value="0.95" />
<characterization id="10" domain="1"
characteristic="4" value="0.55" />
<characterization id="11" domain="2"
characteristic="4" value="0.6" />
<characterization id="12" domain="3"
characteristic="4" value="0.875" />
<characterization id="13" domain="1"
characteristic="5" value="0.675" />
<characterization id="14" domain="2"
characteristic="5" value="0.875" />
<characterization id="15" domain="3"
characteristic="5" value="0.9" />
<characterization id="16" domain="1"
characteristic="6" value="0.8" />
<characterization id="17" domain="2"
characteristic="6" value="0.85" />
<characterization id="18" domain="3"
characteristic="6" value="0.775" />
<characterization id="19" domain="1"
characteristic="7" value="0.7" />
<characterization id="20" domain="2"
characteristic="7" value="0.8" />
```

```
<characterization id="21" domain="3"
characteristic="7" value="0.75" />
<characterization id="22" domain="1"
characteristic="8" value="0.825" />
<characterization id="23" domain="2"
characteristic="8" value="0.85" />
<characterization id="24" domain="3"
characteristic="8" value="0.9" />
</characterization-map>
</meta-model>
<meta-model id="2" name="emotional" type="0">
<domains>
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
<characteristics>
<characteristic id="9" name="third agents representation">third agents
representation</characteristic>
<characteristic id="10" name="emotional cognition">emotional
cognition
</characteristic>
<characteristic id="11" name="goal based behavior">goal based behavior
</characteristic>
<characteristic id="12"
name="agent interaction through sensors and effectors">agent interaction
through sensors and effectors
</characteristic>
<characteristic id="13"
name="agent environment interaction sensibility">agent environment
interaction sensibility
</characteristic>
<characteristic id="14" name="drama simulation skill">drama simulation
```

```
skill
</characteristic>
<characteristic id="15" name="dialog-based skill">dialog-based
skill
</characteristic>
<characteristic id="16" name="dynamic goal generation skill">dynamic goal
generation
skill</characteristic>
</characteristics>
<characterization-map>
<characterization id="10" domain="1"
characteristic="9" value="0.725" />
<characterization id="11" domain="2"
characteristic="9" value="0.875" />
<characterization id="12" domain="3"
characteristic="9" value="0.695" />
<characterization id="13" domain="1"
characteristic="10" value="0.9" />
<characterization id="14" domain="2"
characteristic="10" value="0.525" />
<characterization id="15" domain="3"
characteristic="10" value="0.4" />
<characterization id="16" domain="1"
characteristic="11" value="0.855" />
<characterization id="17" domain="2"
characteristic="11" value="0.895" />
<characterization id="18" domain="3"
characteristic="11" value="0.7" />
<characterization id="19" domain="1"
characteristic="12" value="0.9" />
<characterization id="20" domain="2"
characteristic="12" value="0.475" />
<characterization id="21" domain="3"
characteristic="12" value="0.525" />
<characterization id="4" domain="1" characteristic="13"
value="0.95" />
<characterization id="5" domain="2" characteristic="13"
value="0.725" />
<characterization id="6" domain="3" characteristic="13"
```

```
value="0.625" />
<characterization id="7" domain="1" characteristic="14"
value="0.975" />
<characterization id="8" domain="2" characteristic="14"
value="0.325" />
<characterization id="9" domain="3" characteristic="14"
value="0.375" />
<characterization id="1" domain="1" characteristic="15"
value="0.85" />
<characterization id="2" domain="2" characteristic="15"
value="0.7" />
<characterization id="3" domain="3" characteristic="15"
value="0.725" />
<characterization id="22" domain="1"
characteristic="16" value="0.85" />
<characterization id="23" domain="2"
characteristic="16" value="0.8" />
<characterization id="24" domain="3"
characteristic="16" value="0.825" />
</characterization-map>
</meta-model>
<meta-model id="3" name="Competence" type="1">
<domains>
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
<characteristics>
<characteristic id="17" name="partially observable">partially
observable
</characteristic>
<characteristic id="18" name="deterministic">deterministic
</characteristic>
```

```
<characteristic id="19" name="sequential actions">sequential
actions
</characteristic>
<characteristic id="20" name="dynamic changes">dynamic changes
</characteristic>
<characteristic id="21" name="continuous time line">continuous time
line
</characteristic>
<characteristic id="22" name="multi-agent support">multi-agent
support
</characteristic>
<characteristic id="23" name="competitive rules">competitive
rules
</characteristic>
<characteristic id="24" name="multi-location">multi-location
</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="17"
value="0.9" />
<characterization id="2" domain="2" characteristic="17"
value="0.6" />
<characterization id="3" domain="3" characteristic="17"
value="0.725" />
<characterization id="4" domain="1" characteristic="18"
value="0.625" />
<characterization id="5" domain="2" characteristic="18"
value="0.7" />
<characterization id="6" domain="3" characteristic="18"
value="0.75" />
<characterization id="7" domain="1" characteristic="19"
value="0.875" />
<characterization id="8" domain="2" characteristic="19"
value="0.8" />
<characterization id="9" domain="3" characteristic="19"
value="0.875" />
<characterization id="10" domain="1"
characteristic="20" value="0.9" />
<characterization id="11" domain="2"
```

```
characteristic="20" value="0.925" />
<characterization id="12" domain="3"
characteristic="20" value="0.975" />
<characterization id="13" domain="1"
characteristic="21" value="0.35" />
<characterization id="14" domain="2"
characteristic="21" value="0.8" />
<characterization id="15" domain="3"
characteristic="21" value="0.9" />
<characterization id="16" domain="1"
characteristic="22" value="0.95" />
<characterization id="17" domain="2"
characteristic="22" value="0.975" />
<characterization id="18" domain="3"
characteristic="22" value="0.975" />
<characterization id="19" domain="1"
characteristic="23" value="0.9" />
<characterization id="20" domain="2"
characteristic="23" value="0.875" />
<characterization id="21" domain="3"
characteristic="23" value="0.9" />
<characterization id="22" domain="1"
characteristic="24" value="0.725" />
<characterization id="23" domain="2"
characteristic="24" value="0.6" />
<characterization id="24" domain="3"
characteristic="24" value="0.75" />
</characterization-map>
</meta-model>
<meta-model id="4" name="Virtual 3D" type="1">
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
```

```
</domains>
<characteristics>
<characteristic id="25" name="stochastic">stochastic
</characteristic>
<characteristic id="26" name="episodic">episodic
</characteristic>
<characteristic id="27" name="semi-dynamic">semi-dynamic
</characteristic>
<characteristic id="28" name="distributed visual port">distributed
visual port
</characteristic>
<characteristic id="29" name="physical interaction simulation">physical
interaction
simulation</characteristic>
<characteristic id="30"
name="preconditions and post-conditions">preconditions and post-conditions
</characteristic>
<characteristic id="31" name="normative">normative
</characteristic>
<characteristic id="32" name="fully-observable">fully-observable
</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="25"
value="0.785" />
<characterization id="2" domain="2" characteristic="25"
value="0.685" />
<characterization id="3" domain="3" characteristic="25"
value="0.888" />
<characterization id="4" domain="1" characteristic="26"
value="0.725" />
<characterization id="5" domain="2" characteristic="26"
value="0.725" />
<characterization id="6" domain="3" characteristic="26"
value="0.795" />
<characterization id="7" domain="1" characteristic="27"
value="0.92" />
<characterization id="8" domain="2" characteristic="27"
value="0.824" />
```

```
<characterization id="9" domain="3" characteristic="27"
value="0.745" />
<characterization id="10" domain="1"
characteristic="28" value="0.935" />
<characterization id="11" domain="2"
characteristic="28" value="0.451" />
<characterization id="12" domain="3"
characteristic="28" value="0.325" />
<characterization id="13" domain="1"
characteristic="29" value="0.965" />
<characterization id="14" domain="2"
characteristic="29" value="0.395" />
<characterization id="15" domain="3"
characteristic="29" value="0.47" />
<characterization id="16" domain="1"
characteristic="30" value="0.94" />
<characterization id="17" domain="2"
characteristic="30" value="0.855" />
<characterization id="18" domain="3"
characteristic="30" value="0.835" />
<characterization id="19" domain="1"
characteristic="31" value="0.745" />
<characterization id="20" domain="2"
characteristic="31" value="0.815" />
<characterization id="21" domain="3"
characteristic="31" value="0.758" />
<characterization id="22" domain="1"
characteristic="32" value="0.891" />
<characterization id="23" domain="2"
characteristic="32" value="0.785" />
<characterization id="24" domain="3"
characteristic="32" value="0.698" />
</characterization-map>
</meta-model>
<meta-model id="5" name="Rules based" type="2">
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
```

```
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
<characteristics>
<characteristic id="33" name="basic rule language">basic rule language
</characteristic>
<characteristic id="34" name="communication timeouts">communication
timeouts
</characteristic>
<characteristic id="35" name="delimited messages">delimited
messages
</characteristic>
<characteristic id="36" name="broadcast communication">broadcast
communication
</characteristic>
<characteristic id="37" name="meeting messages">meeting messages
</characteristic>
<characteristic id="38" name="non response penalization">non response
penalization</characteristic>
<characteristic id="39" name="answer-response sequences">answer-response
sequences</characteristic>
<characteristic id="40" name="knowledge rule transmission">knowledge rule
transmission</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="33"
value="0.95" />
<characterization id="2" domain="2" characteristic="33"
value="0.885" />
<characterization id="3" domain="3" characteristic="33"
value="0.89" />
<characterization id="4" domain="1" characteristic="34"
value="0.745" />
<characterization id="5" domain="2" characteristic="34"
value="0.955" />
<characterization id="6" domain="3" characteristic="34"
```

```
value="0.905" />
<characterization id="7" domain="1" characteristic="35"
value="0.565" />
<characterization id="8" domain="2" characteristic="35"
value="0.435" />
<characterization id="9" domain="3" characteristic="35"
value="0.395" />
<characterization id="10" domain="1"
characteristic="36" value="0.855" />
<characterization id="11" domain="2"
characteristic="36" value="0.745" />
<characterization id="12" domain="3"
characteristic="36" value="0.745" />
<characterization id="13" domain="1"
characteristic="37" value="0.69" />
<characterization id="14" domain="2"
characteristic="37" value="0.895" />
<characterization id="15" domain="3"
characteristic="37" value="0.784" />
<characterization id="16" domain="1"
characteristic="38" value="0.905" />
<characterization id="17" domain="2"
characteristic="38" value="0.645" />
<characterization id="18" domain="3"
characteristic="38" value="0.705" />
<characterization id="19" domain="1"
characteristic="39" value="0.72" />
<characterization id="20" domain="2"
characteristic="39" value="0.91" />
<characterization id="21" domain="3"
characteristic="39" value="0.89" />
<characterization id="22" domain="1"
characteristic="40" value="0.9" />
<characterization id="23" domain="2"
characteristic="40" value="0.89" />
<characterization id="24" domain="3"
characteristic="40" value="0.74" />
</characterization-map>
</meta-model>
```

```
<meta-model id="6" name="Dialog based" type="2">
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
<characteristics>
<characteristic id="41" name="minimal set of words ">minimal set of
words
</characteristic>
<characteristic id="42" name="dialog timeout">dialog timeout
</characteristic>
<characteristic id="43" name="multiple listeners dialog">multiple
listeners dialog
</characteristic>
<characteristic id="44" name="idea storm kind dialog">idea storm kind
dialog
</characteristic>
<characteristic id="45"
name="emotional context features dialog">emotional context features dialog</characteris
<characteristic id="46" name="observable dialog feature">observable
dialog feature
</characteristic>
<characteristic id="47" name="secret dialog feature">secret dialog
feature
</characteristic>
<characteristic id="48" name="dialog-based knowledge share">dialog-based
knowledge
share</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="41"
value="0.72" />
<characterization id="2" domain="2" characteristic="41"
```

```
value="0.92" />
<characterization id="3" domain="3" characteristic="41"
value="0.75" />
<characterization id="4" domain="1" characteristic="42"
value="0.812" />
<characterization id="5" domain="2" characteristic="42"
value="0.95" />
<characterization id="6" domain="3" characteristic="42"
value="0.925" />
<characterization id="7" domain="1" characteristic="43"
value="0.86" />
<characterization id="8" domain="2" characteristic="43"
value="0.901" />
<characterization id="9" domain="3" characteristic="43"
value="0.68" />
<characterization id="10" domain="1"
characteristic="44" value="0.88" />
<characterization id="11" domain="2"
characteristic="44" value="0.25" />
<characterization id="12" domain="3"
characteristic="44" value="0.45" />
<characterization id="13" domain="1"
characteristic="45" value="0.86" />
<characterization id="14" domain="2"
characteristic="45" value="0.56" />
<characterization id="15" domain="3"
characteristic="45" value="0.61" />
<characterization id="16" domain="1"
characteristic="46" value="0.91" />
<characterization id="17" domain="2"
characteristic="46" value="0.89" />
<characterization id="18" domain="3"
characteristic="46" value="0.78" />
<characterization id="19" domain="1"
characteristic="47" value="0.78" />
<characterization id="20" domain="2"
characteristic="47" value="0.77" />
<characterization id="21" domain="3"
characteristic="47" value="0.79" />
```

```
<characterization id="22" domain="1"
characteristic="48" value="0.61" />
<characterization id="23" domain="2"
characteristic="48" value="0.7" />
<characterization id="24" domain="3"
characteristic="48" value="0.65" />
</characterization-map>
</meta-model>
<meta-model id="7" name="Hierarchical" type="3">
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
<characteristics>
<characteristic id="49" name="hierarchy defined rules">hierarchy
defined rules
</characteristic>
<characteristic id="50" name="hierarchy levels">hierarchy levels
</characteristic>
<characteristic id="51" name="hierarchy allowed actions">hierarchy
allowed actions
</characteristic>
<characteristic id="52" name="hierarchy roles">hierarchy roles
</characteristic>
<characteristic id="53" name="hierarchy same level groups">hierarchy same
level
groups</characteristic>
<characteristic id="54" name="hierarchy upgrade level rules">hierarchy
upgrade level
rules</characteristic>
<characteristic id="55" name="communication permissions ">communication
permissions </characteristic>
<characteristic id="56" name="service provider reference">service provider
```

```
reference</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="49"
value="0.905" />
<characterization id="2" domain="2" characteristic="49"
value="0.815" />
<characterization id="3" domain="3" characteristic="49"
value="0.885" />
<characterization id="4" domain="1" characteristic="50"
value="0.75" />
<characterization id="5" domain="2" characteristic="50"
value="0.89" />
<characterization id="6" domain="3" characteristic="50"
value="0.405" />
<characterization id="7" domain="1" characteristic="51"
value="0.895" />
<characterization id="8" domain="2" characteristic="51"
value="0.91" />
<characterization id="9" domain="3" characteristic="51"
value="0.7" />
<characterization id="10" domain="1"
characteristic="52" value="0.9" />
<characterization id="11" domain="2"
characteristic="52" value="0.92" />
<characterization id="12" domain="3"
characteristic="52" value="0.651" />
<characterization id="13" domain="1"
characteristic="53" value="0.901" />
<characterization id="14" domain="2"
characteristic="53" value="0.8" />
<characterization id="15" domain="3"
characteristic="53" value="0.59" />
<characterization id="16" domain="1"
characteristic="54" value="0.934" />
<characterization id="17" domain="2"
characteristic="54" value="0.812" />
<characterization id="18" domain="3"
characteristic="54" value="0.671" />
```

```
<characterization id="19" domain="1"
characteristic="55" value="0.889" />
<characterization id="20" domain="2"
characteristic="55" value="0.875" />
<characterization id="21" domain="3"
characteristic="55" value="0.7" />
<characterization id="22" domain="1"
characteristic="56" value="0.65" />
<characterization id="23" domain="2"
characteristic="56" value="0.7" />
<characterization id="24" domain="3"
characteristic="56" value="0.9" />
</characterization-map>
</meta-model>
<meta-model id="8" name="Group organization" type="3">
<domains>
<domain id="1" name="World Simulation">This domain covers the simulation
area.Generally virtual worlds, virtual drama, physical simulations,etc
</domain>
<domain id="2" name="General Business">This domain comprises the business
field with the sell-buy and transaction details and features.</domain>
<domain id="3" name="Sales context">The Sales Context domain covers the sell
and buy features, including auctions, offer and demand laws, special offers,
etc.</domain>
</domains>
<characteristics>
<characteristic id="57" name="membership rules">membership rules
</characteristic>
<characteristic id="58" name="group registry">group registry
</characteristic>
<characteristic id="59" name="group evaluation">group evaluation
</characteristic>
<characteristic id="60" name="group knowledge">group knowledge
</characteristic>
<characteristic id="61" name="group constraints">group constraints
</characteristic>
<characteristic id="62" name="membership protocol">membership protocol
</characteristic>
<characteristic id="63" name="group members roles">group members roles
```

```
</characteristic>
<characteristic id="64" name="group goals">group goals
</characteristic>
</characteristics>
<characterization-map>
<characterization id="1" domain="1" characteristic="57"
value="0.815" />
<characterization id="2" domain="2" characteristic="57"
value="0.545" />
<characterization id="3" domain="3" characteristic="57"
value="0.628" />
<characterization id="4" domain="1" characteristic="58"
value="0.79" />
<characterization id="5" domain="2" characteristic="58"
value="0.651" />
<characterization id="6" domain="3" characteristic="58"
value="0.455" />
<characterization id="7" domain="1" characteristic="59"
value="0.49" />
<characterization id="8" domain="2" characteristic="59"
value="0.75" />
<characterization id="9" domain="3" characteristic="59"
value="0.65" />
<characterization id="10" domain="1"
characteristic="60" value="0.855" />
<characterization id="11" domain="2"
characteristic="60" value="0.6" />
<characterization id="12" domain="3"
characteristic="60" value="0.595" />
<characterization id="13" domain="1"
characteristic="61" value="0.789" />
<characterization id="14" domain="2"
characteristic="61" value="0.645" />
<characterization id="15" domain="3"
characteristic="61" value="0.785" />
<characterization id="16" domain="1"
characteristic="62" value="0.72" />
<characterization id="17" domain="2"
characteristic="62" value="0.89" />
```

```
<characterization id="18" domain="3"
characteristic="62" value="0.5" />
<characterization id="19" domain="1"
characteristic="63" value="0.9" />
<characterization id="20" domain="2"
characteristic="63" value="0.815" />
<characterization id="21" domain="3"
characteristic="63" value="0.566" />
<characterization id="22" domain="1"
characteristic="64" value="0.888" />
<characterization id="23" domain="2"
characteristic="64" value="0.955" />
<characterization id="24" domain="3"
characteristic="64" value="0.85" />
</characterization-map>
</meta-model>
</meta-models>
</MM-Characterization-KB>
```

Nomenclature

AC Agent Centered

ACMAS Agent Centered MAS

ADELFE Atelier de Développement de Logiciels à Fonctionnalité Emergente

ADMACA Application Description Micro-array Characterization

AEIO Agent Environnement Interaction Organisation

AI Artificial Intelligence

AMAS Adaptive Multi-Agent Systems

ANUIES Asociación Nacional de Universidades e Instituciones de Educación Superior

AOSE Agent Oriented Software Engineering

AS Application Specification

AS-Characterization Application Specification Characterization

AS-micro-array Application Specification micro-array

AUML Agent Unified Modelling Language

BCA Bayesian Cognition Algorithm

CI Confidence Interval

CINVESTAV Centro de Investigación y Estudios Avanzados

COSY Systèmes Complexes Coopérants

DESIRE DEsign and Specification of Interacting REasoning framework

DIAMOND Decentralized Iterative Approach for Multiagent Open Networks Design

DMS Decision Making Systems

DSS Decision Support Systems

ECOS Evaluation-orientation de la COopération Scientifique

FIPA Foundation for Intelligent Physical Agent

IPN Instituto Politécnico Nacional

JADE Java Agent DEvelopment Framework

KB Knowledge base

KBs Knowledge bases

LCIS Laboratoire de Conception et Integration de Systèmes

MAMOSACO Méthode Adaptable de Modélisation de Systèmes Administratifs COmplexes

MAS Multi-Agent Systems

MAS Multi-agent systems

MASSIVE MultiAgent SystemS Iterative View Engineering

MDE Model Driven Engineering

MESSAGE Methodology for Engineering Systems of Sotware AGEnts

METALISM METa-Analyse pour L'Ingénierie de software en Systèmes Multi-Agent

MM Meta-Model

MM-Characterization Meta-model Characterization

NCS Non Cooperative Situations

OC Organization Centered

OCMAS Organization Centered MAS

OR Odds Ratio

OSGi Open Services Gateway Initiative

PASSI Process for Agent Societies Specification and Implementation

RR Risk Ratio

RUP Rational Unified Process

UML Unified Modelling Language

XML eXtended Markup Language

Bibliography

20th Century-Fox (2004). The day after tomorrow.

Abrougui, A., Mercier, A., Occello, M., and Huget, M.-P. (2009). Recursive multi-agent system for dynamic and adaptative web services composition. In Chbeir, R., Badr, Y., Kapetanios, E., and Traina, A. J. M., editors, *MEDES*, pages 295–299. ACM.

Adam, E. (2005). *Modèle d'organisation multi-agent pour l'aide au travail coopératif dans les processus d'entreprise : application aux systèmes administratifs complexes*. PhD thesis, Université de Valenciennes et du Hainaut-Cambresis.

Agency, E. S. (2012). Esa - exomars robotic exploration of mars. *http://exploration.esa.int/science-e/www/object/index.cfm?fobjectid=46048*.

Armitage, P., Berry, G., and Matthews, J. (2002). *Statistical methods in medical research*. Blackwell Science.

Baeijs, C. (1998). *Fonctionnalité émergente dans une société d'agents autonomes*. PhD thesis, Institut National Polytechnique de Grenoble.

Baeijs, C. and Demazeau, Y. (1996). Les organisations dans les systèmes multi-agents. In *Proceedings of the Journée Nationale du PRC-IA sur les Systèmes Multi-Agents*, pages 35–46, Nancy, France.

Barthés, J.-P., Chevaillier, P., Courdier, R., Guessoum, Z., Gutknecht, O., Mathieu, P., and Occello, M. (2002). *Organisation et applications des SMA*. Hermes science publications.

Beck, J. C. (1994). A schema for constraint relaxation with instantiations for partial constraint satisfaction and schedule optimization. Master's thesis, University of Toronto.

Beekman, M., Sword, G., and Simpson, S. (2008). Biological foundations of swarm intelligence. *Natural Computing Series*, pages 3–41.

Bernon, C., Camps, V., Gleizes, M.-P., and Picard, G. (2005.). Engineering adaptive multi-agent systems: The adelfe methodology. *Agent-Oriented Methodologies. Idea Group Pub*, pages 172–202.

Bernon, C., Cossentino, M., Gleizes, M.-P., Turci, P., and Zambonelli, F. (2004). A study of some multi-agent meta-models. *Fifth International Workshop on Agent-Oriented Software Engineering (AOSE'04) at AAMAS'04*, pages 62–77.

Bernon, C., Gleizes, M. P., Peyruqueou, S., and Picard, G. (2002a). Adelfe: A methodology for adaptive multi-agent systems engineering. In *ESAW*, pages 156–169.

Bernon, C., Glize, P., Picard, G., and Glize, P. (2002b). The adelfe methodology for an intranet system design. In *Procs. Agent-Oriented Information Systems 2002*, pages 27–28.

Bessière, P., Laugier, C., and Siegwart, R. (2008). *Probabilistic Reasoning and Decision Making in Sensory-Motor Systems*. Springer-Verlag.

Bilisoly, R. (2008). *Practical Text Minning with Perl*. Wiley.

Blumohr, U., Munch, M., and Ukalovic, M. (2011). *Variant Configuration with SAP*. Galileo Press, Incorporated.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. Santa Fe Institute studies in the sciences of complexity. Oxford University Press.

Brahmi, Z. and Gammoudi, M. M. (2009). Hierarchical organization of agents based on galois sub-hierarchy for complex tasks allocation in massive mas. In *Proceedings of the Third KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, KES-AMSTA '09, pages 460–470, Berlin, Heidelberg. Springer-Verlag.

Brazier, F. M. T., Dunin-keplicz, B. M., and Jennings, N. R. (1997). Desire: Modelling multi-agent systems in a compositional formal framework. *International Journal of Cooperative Information Systems*, 6:67–94.

Burrafato, P. and Cossentino, M. (2002). Designing a multi-agent solution for a bookstore with the passi methodology. In Giorgini, P., Lespérance, Y., Wagner, G., and Yu, E. S. K., editors, *AOIS@CAiSE*, volume 57 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Caire, G., Coulier, W., Garijo, F., Gomez, J., Pavon, J., Leal, F., Chainho, P., Kearney, P., Stark, J., Evans, R., Massonet, P., and Park, B. A. (2001a). Agent oriented analysis using message/uml. In *Revised Papers and Invited Contributions from the Second International Workshop on Agent-Oriented Software Engineering II*, pages 119–135. Springer-Verlag.

Caire, G., Garijo, F., Gomez, J., Pavon, J., and Vargas, E. (2001b). Agent oriented analysis using message/uml. In *AOSE*.

Camps, V. and Gleizes, M.-P. (1995). Principes et évaluation d'une méthode d'auto-organisation. In *Actes des troisièmes journées francophones IADSMA, St Baldoph-Savoie, 15/03/95-17/03/95*, pages 337–348. -.

Castro, J., Giorgini, P., Kolp, M., and Mylopoulos, J. (2005). Tropos: A requirements-driven methodology for agent-oriented software. *Agent Oriented Methodologies*, pages pp 20–45.

Chandrasekaran, B., Josephson, J. R., and Benjamins, V. R. (1999). What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14:20–26.

Corning, P. A. (1998). Complexity is just a word! web paper.

Cossentino, M. (2005). From requirements to code with the passi methodology. *Agent-Oriented Methodologies*, pages pp.79–106.

Cossentino, M., Gleizes, M.-P., Molesini, A., and Omicini, A. (2011). F engineering and aose. In *Proceedings of the 10th international conference on Agent-oriented software engineering*, AOSE'10, pages 191–212, Berlin, Heidelberg. Springer-Verlag.

Dastani, M., Hulstijn, J., Dignum, F., and Meyer, J.-J. C. (2004). Issues in multiagent system development. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '04, pages 922–929, Washington, DC, USA. IEEE Computer Society.

Deloach, S. A. (2001). Analysis and design using mase and agenttool. In *In 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2001)*.

Demazeau, Y. (1993). La Plate-forme PACO et ses Applications. In *2èmes Journée Nationale du PRC-IA sur les Systèmes Multi-Agents, PRC-IA, Montpellier, France*.

Demazeau, Y. (1995.). From interactions to collective behaviour in agent-based systems. *I EuropeanConference on Cognitive Science*.

Demazeau, Y. (2001). *Rapport d'Habilitation a Diriger des Recherches- Agentes Voyelles*. PhD thesis, Institut National Polytechnique de Grenoble, Laboratoire Leibniz.

Di Marzo Serugendo, G., Gleizes, M.-P., and Karageorgos, A. (2006). Self-organisation and emergence in multi-agent systems: An overview. *Informatica*, 30(1):45–54. ISSN 0350-5596.

Dignum, V., Dignum, F., and Sonenberg, L. (2004). Towards dynamic reorganization of agent societies. In *In Proceedings of Workshop on Coordination in Emergent Agent Societies*, pages 22–27.

Drogoul, A. and Collinot, A. (1998a). Applying an agent-oriented methodology to the design of artificial organizations: A case-study in robotic soccer. *Autonomous Agents and Multi-Agent Systems*, 1:113–129.

Drogoul, A. and Collinot, A. (1998b). Using the cassiopeia method to design a soccer robot team. *Applied Artificial Intelligence: An International Journal*, 2 and 3:127–147.

Evans, E. (2003). *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison Wesley.

Federico Bergenti, Marie-Pierre Gleizes, F. Z., editor (2004). *Methodologies and Software Engineering for Agent Systems*. Springer.

Ferber, J. and Gutknecht, O. (1997). Aalaadin: A meta-model for the analysis and design of organizations in multi-agent systems.

Ferber, J. and Gutknecht, O. (1998a). A meta-model for the analysis and design of organizations in multi-agent systems. In *Proceedings of the 3rd International Conference on Multi Agent Systems*, ICMAS '98, pages 128–, Washington, DC, USA. IEEE Computer Society.

Ferber, J. and Gutknecht, O. (1998b). A meta-model for the analysis and design of organizations in multi-agent systems. In Demazeau, Y., editor, *Proceedings of ICMAS'98*, Paris France. IEEE Computer Society.

Ferber, J. and Gutknecht, O. (1998c). A meta-model for the analysis and design of organizations in multi-agent systems. In *ICMAS '98: Proceedings of the 3rd International Conference on Multi Agent Systems*, pages 128–135, Washington, DC, USA. IEEE Computer Society.

Ferber, J., Gutknecht, O., and Michel, F. (2003). From agents to organizations: An organizational view of multi-agent systems. In *In: LNCS n. 2935: Procs. of AOSE 03*, pages 214–230. Springer Verlag.

FIPA (2002). Fipa contract net interaction protocol specification-no. sc00029h. *Online - FIPA*, page [Online] Retrieved: http://www.fipa.org/specs/fipa00029/SC00029H.pdf.

Fox, M. S. (1988). An organizational view of distributed systems. In Bond, A. H. and Gasser, L., editors, *Distributed Artificial Intelligence*, pages 140–150. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Francis, J. B. and Heylighen, F. (1996). Algorithms for the self-organisation of distributed, multi-user networks. possible application to the future world wide web. In *World Wide Web, in: Cybernetics and Systems '96 R. Trappl (ed.), (Austrian Society for Cybernetics*, pages 911–916.

Frederic, G. (1996). *La réorganisation dynamique dans les systèmes multi-agents*. PhD thesis, University of Savoie.

Garcia-Ojeda, J. C., Perez-Alcazar, J. d. J., and Arenas, A. E. (2004). Extending the gaia methodology with agent-uml. *AAMAS*, pages 1456–1457.

Gasevic, D., Djuric, D., and Devedzic, V. (2009a). *Model Driven Engineering and Ontology Development*. Springer-Verlag.

Gasevic, D., Djuric, D., and Devedzic, V. (2009b). *Model Driven Engineering and Ontology Development*. Springer-Verlag.

Giorgini, J. Mylopoulos, M. P. (2004). The tropos methodology: an overview. *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, page 505.

Giret, A. and Botti, V. J. (2004). On the definition of meta-models for analysis of large-scale mas. In *MATES*, pages 273–286.

Glansdorff, P. and Prigogine, I. (1971). *Thermodynamic theory of structure, stability and fluctuations*. Wiley-Interscience.

Gómez, C., Isern, D., and Moreno, A. (2007). Software engineering methodologies to develop multi-agent systems : State of the art. Technical report, Department of Computer Science and Mathematics, Universitat Rovira i Virgili.

Google (2011). Google translator. online tool.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowl. Acquis.*, 5:199–220.

Guttman, R. H. and Maes, P. (1998). Cooperative vs. competitive multi-agent negotiations in retail electronic commerce. In *Proceedings of the Second International Workshop on Cooperative Information Agents (CIA'98*, pages 135–147.

Hasselblatt, B. and Katok, A. (2003). *A first course in dynamics: with a panorama of recent developments*. Cambridge University Press.

Hendler, J. (2001). Agents and the semantic web. *IEEE Intelligent Systems*, 16:30–37.

Holland, J. (1995). *Hidden Order: How Adaptation Builds Complexity*. Helix Books.

Holland, J. (1998). *Emergence: From Chaos to Order*. Oxford University Prees.

Humphrys, M. (1995). Towards self-organising action selection.

I. Jacobson, G. Booch, J. R. (1999). *The Unified Software Development Process*. Addison-Wesley,.

Ishida, T., Gasser, L., and Yokoo, M. (1992). Organization self-design of distributed production systems. *IEEE Transactions on Knowledge and Data Engineering*, 4:123–134.

ITAR-TASS (2012). Phobos-grunt chips supposedly were counterfeit. *http://www.itar-tass.com/en/c32/330734.html*.

J. Odell, H.V. Parunak, B. B. (2000). Extending uml for agents. In *Agent Oriented Information Systems (AOIS) Workshop*. 17th National Conference on Artificial Intelligence (AAAI).

Jamont, J.-P. (2005). *DIAMOND : Une approche pour la conception de systèmes multi-agents embarqués*. PhD thesis, Institut National Polytechnique de Grenoble (INP Grenoble).

Jamont, J.-P. and Occello, M. (2007). Designing embedded collective systems: The diamond multiagent method. In *ICTAI (2)*, pages 91–94.

Jamont, J.-P., Occello, M., and Lagrèze, A. (2010). A multiagent approach to manage communication in wireless instrumentation systems. *Measurement*, 43(4):489 – 503.

Jennings, N. R., Norman, T. J., and Faratin, P. (1998). Adept: An agent-based approach to business process management. *ACM SIGMOD Record*, 27:32–39.

Joann Roskoski, Jeannette Wing, J. M. n. (2003). Cyberinfrastructure framework for 21st century science and engineering. *Online - National Science Foundation*, page [Online] Retrieved: http://www.nsf.gov/pubs/2010/nsf10015/nsf10015.jsp.

Johnson, S. (2002). *Emergence: the connected lives of ants, brains, cities, and software*. Simon & Schuster.

Jorge Gómez-Sanz, J. P. (2004). Methodologies for developing multi-agent systems. *Journal of Universal Computer Science*, 10:359–374.

Joslyn, C. and Rocha, L. M. (2000). Towards semiotic agent-based models of socio-technical organizations. In *in: Proc. AI and Simulation 2000, ed. HS Sarjoughian et al*, pages 70–79.

Jürgen, L. (2001). *Iterative software engineering for multiagent systems: the MASSIVE method*. Springer-Verlag, Berlin, Heidelberg.

Kelvin, B. W. T. (1892). *The kinetic theory of the dissipation of energy*. Cambridge University Press.

Kelvin, S. W. T. (1879). The sorting demon of maxwell. *Proceedings of the Royal Institution*, IX:113.

Kleppe, A., Warmer, J., and Bast, W. (2003). *MDA Explained: The Model Driven Architecture: Practice and promise*. Addison Wesley.

Knublauch, H. and Rose, T. (2002). Tool-supported process analysis and design for the development of multi-agent systems. In *In Proc. of the Third Int. Workshop on Agent-Oriented Software Engineering (AOSE-02*.

Koehn, P. (2009). *Statistical Machine Translation*. Cambridge University Press.

Laplace, P.-S. (1812). *Essai philosophique sur les probabilités*. Edition Veuve Courcier.

LaPorte, R. (2011). Course: How to conduct a meta-analysis.

Leandro, G. (2005). *Meta-analysis in Medical Research,The handbook for the understanding and practice of meta-analysis*. BMJ Books.

Lind Padgham, M. W. (2004). *Developing Intelligent Agent Systems a practical guide*. Jhon Wiley & Sons, Ltd.

Luck, M., McBurney, P., Shehory, O., and Willmott., S. (2005). *Agent technology: Computing as interaction (a roadmap for agent based computing)*. University of Southampton.

Maes, P. (1994). Agents that reduce work and information overload. *Commun. ACM*, 37:30–40.

Malone, T. (1987). Modeling coordination in organizations and markets. In *Management science*, volume 33, pages 1317–1332.

Marcelpoil, R., Beaurepaire, E., and Pesty, S. (1994). La sociologie cellulaire : modéliser et simuler une "société cellulaire" pour étudier le vivant. *Intellectica*, 19:53–72.

Marcenac, P., Courdier, R., and Souli, J. C. (1998). Towards an emergence machine for complex systems simulations. In *Lecture Notes in Artificial Intelligence*, pages 785–795. Springer Verlag.

MARCIA-(Group) (1996). Auto-organisation := évolution de structure(s). In *Actes des 4ème journées Nationales du PRC-IA sur les systèmes Multi-Agents (PRC-IA)*, Toulouse, France.

Mariachiara, P., Cossentino, M., Cabri, G., and Molesini, A. (2010). Building an agent methodology from fragments: the mensa experience. In *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 920–927, New York, NY, USA. ACM.

Massimo, V. S., Cossentino, M., and Gaglio, S. (2006). A repository of fragments for agent systems design. In *Proc. Of the Workshop on Objects and Agents (WOA06)*, pages 130–137.

Mercier, A. and Beigbeder, M. (2005). Extraction de la localisation des termes pour le classement des documents. In *EGC*, pages 275–280.

Merriam-Webster, I. S. (1981). *Webster's new collegiate dictionary*. G. & C. Merriam Co.

Michael (2002). *An Introduction to MultiAgent Systems*. Jhon Wiley & Sons, Ltd.

Milidiu, R. L., Melcop, T., Liporace, F. d. S., and Lu, C. J. P. d. (2003). Simple - a multi-agent system for simultaneous and related auctions. In *Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology*, IAT '03, pages 511–, Washington, DC, USA. IEEE Computer Society.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Commun. ACM*, 38:39–41.

Minsky, M. (1986). *The society of mind*. Simon & Schuster, Inc., New York, NY, USA.

Mintzberg, H. (1979). *The structuring of organizations: a synthesis of the research*. Theory of management policy series. Prentice-Hall.

MIT (2009). Mit technology review 2009 special report. online.

Muscettola, N., Nayak, P. P., Pell, B., and Williams, B. C. (1998). Remote agent: To boldly go where no ai system has gone before.

NASA (2011). Nasa in situ resource utilization. *http://isru.msfc.nasa.gov/*.

New-Line-Cinema (2001). The lord of the rings special effects feature.

NIH (2010). Pubmed online. *U.S. National Library of Medicine*, page [online] http://www.ncbi.nlm.nih.gov/pubmed/.

O. Gutknecht, J. F. (1999). Vers une méthodologie organisationnelle pour les systèmes multi-agents. *Actes des Journées Francophones d'Intelligence Artificielle Distribuée et Systèmes Multi-Agents, Saint-Denis, Réunion*.

Occello, M. (2000). Mirage: a generic model to build recursive multiagent systems. *International Conference on Artificial Intelligence.*

Occello, M., Baeijs, C., Demazeau, Y., and Koning, J.-L. (2004). Mask: An aeio toolbox to design and build multi-agent systems. In et al, C., editor, *Knowledge Engineering and Agent Technology*, IOS Series on Frontiers in AI and Applications.

OMG (2003). Unified modeling language (uml) specification: Infrastructure, version 2.0, final adopted specification,.

P. Giorgini, J. Mylopoulos, M. P. (2004). The tropos methodology: an overview. *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, page p. 505.

Padgham, L. and Winikoff, M. (2004). The prometheus methodology. In *Methodologies and Software Engineering for Agent Systems, chapter 11*. Kluwer Academic Publishing (New.

Parsons, S., Wooldridge, M., and Amgoud, L. (2002). An analysis of formal inter-agent dialogues. In *In 1st International Conference on Autonomous Agents and Multi-Agent Systems*, pages 394–401. ACM Press.

Pavón, J. and Gómez-Sanz, J. (2002). Agent oriented software engineering with ingenias. In *In Proceedings of Iberagents 2002, Agent Technology and Software Engineering*.

Qiao, B. and Zhu, J. (2011). "agent-based intelligent manufacturing system for the 21st century,". online.

Ramos, F. F., Razo, L., Martinez, A. V., Zúñiga, F., and Piza, H. I. (2005). 3d emotional agent architecture. In Bui, A., Bui, M., Böhme, T., and Unger, H., editors, *IICS*, volume 3908 of *Lecture Notes in Computer Science*, pages 181–194. Springer.

Ray, L. B. (2011). Computer scientists learn from flies. *Sci. Signal.*, 4(156):ec21.

Razo, L., Ramos, F., and Occello, M. (2010). METAOSE: Meta-analysis for agent oriented software engineering. In *Electronics, Robotics and Automotive Mechanics Conference- CERMA 2010 Electronics, Robotics and Automotive Mechanics Conference- CERMA 2010*, pages 197–204, Mexique.

Re, G. and Pius, H. (2003). Distributed decision support systems:. In *in R. Bisdorff (eds) Human Centered Processes - Distributed Decision Making and ManMachine Cooperation, proceedings of the 14th MINI EURO Conference*, pages 211–216.

Reilly, W. S., Carbonell, J., Simmons, R., Scott, W., Scott, W., Reilly, N., and Reilly, N. (1996). Believable social and emotional agents.

Ricordel, P.-M. and Demazeau, Y. (2002). Volcano : a vowels-oriented multi-agent platform. In *Proceedings of the International Conference of Central Eastern Europe on Multi-Agent Systems (CEEMAS'01), From Theory to Practice in Multi-Agent Systems*, LNAI 2296, pages 252–262. Springer Verlag.

Rogers, A., David, E., Jennings, N. R., and Schiff, J. (2007). The effects of proxy bidding and minimum bid increments within ebay auctions. *ACM Trans. Web*, 1.

Roscosmos (2011). Roscosmos - phobos-grunt unmmaned mission. *http://www.roscosmos.ru/main.php?id=376*.

Rougemaille, S., Migeon, F., Maurel, C., and pierre Gleizes, M. (2008). Model driven engineering for designing adaptive multi-agents systems. In *Engineering Societies in the Agents World VIII*, volume 4995 of *Lecture Notes in Computer Science*, pages 318–332. Springer Berlin / Heidelberg.

Russel, S. and Norvig, P. (2003). *Artificial Intelligence: a Modern Approach*. Prentice Hall.

Sánchez, D. and Moreno, A. (2008). Learning non-taxonomic relationships from web documents for domain ontology construction. *Data Knowledge Engineering*, 64:600–623.

Schurr, N., Marecki, J., Tambe, M., and Scerri, P. (2005). The future of disaster response: Humans working with multiagent teams using defacto. In *In AAAI Spring Symposium on AI Technologies for Homeland Security*.

Seidita, V., Cossentino, M., and Gaglio, S. (2006). S.: A repository of fragments for agent systems design. In *Proc. Of the Workshop on Objects and Agents (WOA06*, pages 130–137.

Selic, B. (2003). The pragmatics of model-driven development. *IEEE Softw.*, 20(5):19–25.

Selten, R. (1975). Spieltheoretische behandlung eines oligopolmodells mit nachfragetragheit. *Zeitschrift fur die gesamte Staatswissenschaft*, 121:301–324.

Serugendo, G. D. M., Gleizes, M.-P., and Karageorgos, A. (2004). Agentlink first technical forum group self-organisation in multi-agent systems. *AgentLink Newsletter, Issue 16*.

Shen, W. and Norrie, D. H. (1999). Agent-based systems for intelligent manufacturing: A state-of-the-art survey. *Knowledge Inforgamtion Systems*, pages 129–156.

Shoham, Y. (1993). Agent oriented programming. In North-Holland., editor, *Artificial Intelligence*, volume 60, pages p. 51–92.

Siebers, P.-O. and Aickelin, U. (2008). Introduction to multi-agent simulation. *CoRR*, abs/0803.3905.

Siu-Kwong, Y. E. (1996). *Modelling strategic relationships for process reengineering*. PhD thesis, University of Toronto, Toronto, Ont., Canada, Canada. UMI Order No. GAXNN-02887 (Canadian dissertation).

So, Y. P. and Durfee, E. (1993). An organizational self-design model for organizational change. In *In AAAI93 Workshop on AI and Theories of Groups and Oranizations*, pages 8–15.

Stuart Russell, P. N. (2009). *Artificial Intelligence: A modern approach*. Prentice Hall.

Sun, R. and Naveh, I. (2004). Simulating organizational decision-making using a cognitively realistic agent model.

Sutton, A. J., Jones, D. R., Abrams, K. R., Sheldon, T. A., and Song, F. (2000). *Methods for Meta-analysis in Medical Research.* London: John Wiley.

Thomas, S. (1993). *PLACA, an Agent Oriented Programming Language*. PhD thesis, Stanford University.

TILab (2011). Jade (java agent development framework).

Trianni, V., Nolfi, S., and Dorigo, M. (2008). Evolution, self-organization and swarm robotics. *Natural Computing Series*, pages 162–191.

Troyan, E. (2009). Helium-3 fuel of the future. *http://www.7oon.eu/helium.htm*.

Uckelman, S. L. (2010). Obligationes as formal dialogue systems. *Frontiers in Artificial Intelligence and Applications*, 222:341 – 353.

Vijayan Sugumaran, V. C. S. (2002). An ontology-based framework for generating and improving database design. *Natural Language Processing and Information Systems*.

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards*. PhD thesis, University of Cambridge.

Wei Huang, Elia El-Darzi, L. J. (2007). Extending the gaia methodology for the design and development of agent-based software systems. *COMPSAC*, pages 159–168.

Weiss, G. (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. Intelligent Robotics and Autonomous Agents. MIT Press.

Wooldridge, M. (2000). *Reasoning About Rational Agents*. The MIT Press.

Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*. John Wiley & Sons.

Wooldridge, M., Jennings, N. R., and Kinny, D. (2000). The gaia methodology for agent-oriented analysis and design. *Journal of Autonomous Agents and Multi-Agent Systems*, 3:285–312.

Z.Guessoum and Jarraya, T. (2005). Meta-models and model-driven architectures. *Contribution to the AOSE TFG AgentLink3 meeting*.

Zhang, S., Xuan, P., and Patel, B. (2009). Managing operations in multiagent virtual organizations. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, pages 1309–1310, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Zhang, Z. and David, F. (2005). An introduction to mama (meta-analysis of microarray data) system. *27th Annual International Conference of the Engineering in Medicine and Biology Society*.

Zúñiga, F., Ramos, F. F., and Piza, H. I. (2005). Geda-3d agent architecture. In *ICPADS (2)*, pages 201–205. IEEE Computer Society.