A PROCESS FOR MULTI-AGENT DOMAIN AND APPLICATION ENGINEERING

The Domain Analysis and Application Requirements Engineering Phases

Adriana Leite[1] and Rosario Girardi[2]
[1]*Department of Computer Science, Federal Institute of Maranhão, Avenida Getúlio Vargas, São Luís, Brazil*
[2]*Department of Computer Science, Federal University of Maranhão, Avenida Dos Portugueses, São Luís, Brazil*

Keywords: Software Process, Requirements Engineering, Multi-agent Systems, Reuse, Ontologies.

Abstract: Domain Engineering is a process for the development of a reusable application family in a particular domain problem, and Application Engineering, the one for the construction of a specific application based on the reuse of software artifacts in the application family previously produced in the Domain Engineering process. MADAE-Pro is an ontology-driven process for multi-agent domain and application engineering which promotes the construction and reuse of agent-oriented applications families. This article introduces an overview of MADAE-Pro emphasizing the description of its domain analysis and application requirements engineering phases and showing how software artifacts produced from the first are reused in the last one.

1 INTRODUCTION

A software development process is a model that specifies a life cycle, describing the phases through which transits a software product from its conception through its development along with a methodology that integrates the techniques to be applied in each one of the phases according to a development paradigm.

MADAE-Pro ("Multi-agent Domain and Application Engineering Process") is a process for the development and reuse of families of multi-agent software systems. A family of software systems is defined as a set of systems sharing some commonalities but also having particular features (Czarnecki and Eisenecker, 2000). It consists of two complementary sub-processes: Multi-agent Domain Engineering and Multi-agent Application Engineering.

Multi-agent Domain Engineering is a process for the development of a family of multi-agent software systems in a problem domain, by applying MADEM ("Multi-agent Domain Engineering Methodology") (Girardi, 1992); and Multi-agent Application Engineering, the one for constructing a specific agent-oriented application by reusing one or more of those families, using MAAEM ("Multi-agent Application Engineering Methodology") (Leite et al., 2008). The MADAE-Pro products are represented as facts of the ONTORMAS knowledge base. ONTORMAS ("ONTOlogy driven tool for the Reuse of Multi-Agent Systems") (Leite et al., 2008) is a knowledge-based tool for supporting and automating the MADAE-Pro tasks.

The paper is organized as follows. Section 2 describes the MADAE-Pro software development process. Section 2.1 introduces its lifecycle and a general description of the support that the MADEM and MAAEM methodologies provide to each one of its phases. Section 2.2 details the particular tasks of the Domain Analysis and Application Requirements Engineering phases along with the guidelines provided by these methodologies to carry out those tasks. Section 3 references related work discussing its similarities and differences with MADAE-Pro. Finally, section 4 concludes the paper with some considerations on ongoing work.

2 THE MADAE-Pro PROCESS

MADAE-Pro is a knowledge-based process. Its phases, tasks and products are conceptualized in the ONTORMAS knowledge-base and both, particular or multi-agent system families are represented as instances of this knowledge base.

Main modeling concepts and tasks of MADEM and MAAEM are based both on techniques for

Leite A. and Girardi R. (2009).
A PROCESS FOR MULTI-AGENT DOMAIN AND APPLICATION ENGINEERING - The Domain Analysis and Application Requirements Engineering Phases.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Information Systems Analysis and Specification*, pages 156-161
DOI: 10.5220/0001990201560161

Domain and Application Engineering (Czarnecki and Eisenecker, 2000) and for development of multi-agent systems (Bresciani et al., 2004) (Cossentino et al., 2004) (Dileo,Jacobs and Deloach, 2002). A specific graphical modeling language has been defined for the representation of the MADAE-Pro concepts. For the specification of the problem to be solved, the methodologies focus on modeling goals, roles and interactions of entities of an organization. Entities have knowledge and use it to exhibit autonomous behavior. An organization is composed of entities with general and specific goals that establish what the organization intends to reach. The achievement of specific goals allows reaching the general goal of the organization. Specific goals are reached through the performance of responsibilities that entities have by playing roles with a certain degree of autonomy. Entities playing roles have skills on one or a set of techniques that support the execution of responsibilities in an effective way. Pre-conditions and post-conditions may need to be satisfied for/after the execution of a responsibility. Knowledge can be consumed and produced through the execution of a responsibility.

2.1 The MADAE-Pro Lifecycle

The MADAE-Pro process life cycle is iterative, incremental and goal-driven. Development is carried out through successive increments, looking for reducing software complexity. It is initiated with the decision of development of a new family of applications or a specific one by specifying a new general goal and restarted for the development of a new specific goal or to update an existing one in evolutive and corrective maintenance, respectively. Iterations also can occur between the phases for refining modeling products.

MADAE-Pro consists of six development phases: domain analysis, domain design and domain implementation, supported by the MADEM methodology; and application requirements engineering, application design and application implementation, supported by the MAAEM methodology.

The domain analysis phase of MADEM approaches the construction of a domain model specifying the current and future requirements of a family of applications in a domain by considering domain knowledge and development experiences extracted from domain specialists and applications already developed in the domain. This phase consists of the following modeling tasks: modeling of domain concepts, goal modeling, role modeling, role interaction modeling and user interface prototyping. The product of this phase, a domain model, is obtained through the composition of the products constructed through these tasks: a concept model, a goal model, a role model, a set of role interaction models, one for each specific goal in the goal model and a prototype of the user interface. Next section details the domain analysis tasks and products.

The domain design phase of MADEM approaches the architectural and detailed design of multi-agent frameworks providing a solution to the requirements of a family of multi-agent software systems specified in a domain model. This phase consists of two sub-phases: the architectural design sub-phase which establishes an architectural model of the multi-agent society including the knowledge shared by all agents in their communication and their coordination and cooperation mechanisms; and the agent design sub-phase which defines the internal design of each agent, by modeling its structure and behavior. A Multi-agent Framework Model of the Multi-agent Society is constructed as a product of this phase, composed of a Multi-agent Society Knowledge Model, an Architectural Model and a set of Agent Models.

The domain implementation phase of MADEM approaches the mapping of design models to agents, behaviors and communication acts, concepts involved in the JADE framework (Bellifemine et al., 2003), which is the adopted implementation platform. An implementation model of the multi-agent society is constructed as a product of this phase, composed of a model of agents and behaviors and a model of communication acts.

Variability modeling is carried out in parallel with all MADEM phases to determine the common and variable parts of an application family. This is done by identifying the “Variation Points” and its correspondent “Variants”. A variation point is the representation of a concept subjected to variation. A variant represents the alternative or optional variations of such a concept.

MAAEM is a methodology for requirement analysis, design and implementation of multi-agent applications through compositional reuse of software artifacts such as domain models, multi-agent frameworks, pattern systems and software agents previously developed in the MADEM Domain Engineering process.

The requirements analysis phase of MAAEM looks for identifying and specifying the requirements of a particular application by reusing requirements already specified in domain models. This phase follows a set of modeling tasks consistently uniform with the ones of the MADEM

domain analysis phase, for producing a set of models composing the multi-agent application specification. The MAAEM requirements analysis phase is performed through the following modeling tasks: concept modeling, goal modeling, role modeling, role interaction modeling and user interface prototyping. The product of this phase, an application specification, is obtained through the composition of the products constructed through these tasks: a concept model, a goal model, a role model, a set of role interaction models, one for each specific goal in the goal model and a prototype of the user interface. Next section details the requirements analysis tasks and products.

In the application design phase, developers reuse design solutions of a family of applications and adapt them to the specific requirements of the application under development. A set of models composing the multi-agent application architecture are produced by following a set of modeling tasks consistently uniform with the ones of the MADEM domain design phase. This phase consists of two tasks: the Architectural Design task aiming at constructing a multi-agent society architectural model and the Agent Design task, which defines the internal structure of each agent in the society. The Architectural Design task consists of five sub-tasks: Multi-agent Society Knowledge Modeling, Multi-Agent Society Modeling, Agent Interaction Modeling, Activity Modeling, and Coordination and Cooperation modeling.

In the application implementation phase, agent behaviours and interactions are identified and specified in a particular language/platform for agent development. A Behaviors Model and Communication Acts Model are generated in this development phase.

Along all MAAEM phases, reuse is carried out by identifying variation points in MADEM products and selecting appropriate variants.

2.2 The Domain Analysis and Application Requirements Engineering Tasks

This section describes the Domain Analysis and Application Requirements Engineering tasks of MADAE-Pro showing how software artifacts of the ONTOSERS domain model (Mariano et al., 2008) are produced and reused on the development of the InfoTrib multi-agent recommender system. ONTOSERS-DM is a domain model that specifies the common and variable requirements of recommender systems based on the ontology technology of the Semantic Web, using three information filtering approaches: content-based (CBF), collaborative (CF) and hybrid filtering (HF) (Mariano et al., 2008).

InfoTrib is a Tax Law recommender system in which, based on a user profile specifying his/her interests in the diverse species of taxes, the system provides recommendations based on new tax law information items.

The Domain Analysis Tasks. The modeling of domain application concepts task aims at performing a brainstorming of domain concepts and their relationships, representing them in a concept model.

The purpose of the goal modeling task is to identify the common and variant goals of the family of systems, the external entities with which it cooperates and the responsibilities needed to achieve them. Its product is a goal model, specifying the general and specific goals of the system family along with the external entities, responsibilities and variant groups. In this task, variability modeling looks for identifying variant points in specific goals related with variant groups of responsibilities.

The goal model of ONTOSERS-DM has the "Provide Recommendations using Semantic Web Technology" general goal. It is reached through the "Model Users", "Filter Information" and "Deliver Recommendations" specific goals. In order to achieve the "Filter Information" specific goal, it is necessary to perform the "Ontology Instance User Model Creation and Update" responsibility, which also contributes to reach the "Model Users" specific goal. Besides that, the "Grouping of user models", "Information Items based on Ontology Instance Representation" and "Similarity Analysis" responsibilities are needed. The "Grouping of Users Models" responsibility allows for identifying groups of users with similar interests.

The "Model Users" specific goal has a variation point with groups of responsibilities for user profile acquisition, being possible to choose between three alternative variants: "Implicit Profile Acquisition", "Explicit Profile Acquisition" or both. The last responsibility, "Ontology Instance User Model Creation and Update" is fixed, i.e. it is required in all the applications of the family. The "Filter Information" specific goal has a variation point that has as variant alternatives: the "Grouping of users models" responsibility, required in systems that use CF; and the "Information Items based on Ontology Instance Representation" responsibility required in the ones using CBF. The "Deliver Recommendations" specific goal does not have variation points, therefore the "Similarity Analysis", "Personalized Recommendations Production" and

"Delivery of Personalized Recommendations" responsibilities are required in all the applications of the family, then belonging to the fixed part of the goal model.

The role modeling task associates the responsibilities, either common or variants, identified in the goal modeling task to the roles that will be in charge of them. The pre and post-conditions that must be satisfied before and after the execution of a responsibility are also identified. Finally, the knowledge required from other entities (roles or external entities) for the execution of responsibilities and the knowledge produced from their execution is identified. This task produces a set of role models, one for each specific goal or, having it one or more variation points, one role model for each variant, specifying roles, responsibilities, pre- and post-conditions, knowledge and relationships between these concepts. The role interaction modeling task aims at identifying how external and internal entities should cooperate to achieve a specific goal. For that, responsibilities of roles are analyzed along with their required and produced knowledge specified in a role model. A set of role interaction models is constructed as a product of this modeling task, one for each specific goal, or having it one or more variation points, one role interaction model for each variant, specifying the interactions between roles and external entities needed to achieve a specific goal. The interactions are numbered according to their sequencing. Finally, a User Interface Prototype, is developed by identifying the interactions of users with the system family.

The Application Requirements Engineering Tasks. In this phase, the reuse of domain models is supported by the ONTORMAS tool. In ONTORMAS, the selection of software artifacts is supported by semantic retrieval, where the user inputs a query specifying the product features he/she intends to reuse and gets as result the available artifacts from the repository which satisfies his query. After the selection of the artifact that most closely matches their needs, the user should check if the artifact can be integrally reused or if it needs adaptations and/or integrations with other artifacts.

The modeling of application concepts task aims at performing a brainstorming of the application concepts and their relationships, representing them in a concept model. The purpose of the goal modeling task is to identify the goals of the application, the external entities with which it cooperates and the responsibilities needed to achieve them. Its product is a goal model, specifying the general and specific goals of the application along with the external entities and responsibilities. This task should be reuse-intensive. From the concept model and from a first draft of the goal model, possible terms for searching and reusing goals in already available domain models can be revealed. If a general goal is identified, the corresponding goal model in a domain model is selected for reuse. If a specific goal is identified, this goal, sub-goals in a possible hierarchy, related responsibilities and external entities in a goal model of a domain model are selected for reuse. Otherwise, the goal model is constructed from scratch.

If a selected specific goal or sub-goals in its hierarchy have associated variation points, they should be analyzed to select and possible reuse the appropriate variants of alternative or optional groups of responsibilities. Only one group of responsibilities in an alternative variant can be selected for reuse. Zero or more groups of responsibilities in an optional variant can be selected for reuse.

To construct the goal model of InfoTrib, first, a search in ONTORMAS with the term "recommendation" was done. The general goal "Provide recommendations using semantic web technologies" was retrieved through the search. Therefore, the corresponding goal model was selected for reuse, in this case, the goal model of ONTOSERS, part of the ONTOSERS domain model. From the variation point of the "Model users" specific goal, the "Explicit profile acquisition" responsibility variant was selected in order to support just the explicit acquisition of user profiles. From the variation point of the "Filter Information" specific goal, the "Information Items based on Ontology Instance Representation" responsibility variant was selected for providing content-based information filtering. The names of the external entities "User" and "Ontology based information source" were specialized to "Tributary User" and "ONTOTRIB", the ontology that defines the Tax Law concepts and relationships.The role modeling task associates the responsibilities identified in the goal modeling task to the roles that will be in charge of them. The pre and post-conditions that must be satisfied before and after the execution of a responsibility are also identified. Finally, the knowledge required from other entities (roles or external entities) for the execution of responsibilities and the knowledge produced from their execution is identified. A set of role models, one for each specific goal in the goal model is constructed in this task, with or without reuse.The following rules apply for the reuse activities performed during this modeling task:

• If a similar general goal was identified during the goal modeling task, thus reusing fully or partially a goal model then, the set of role models, already available in the corresponding domain model and associated to each reused specific goal will be reused and eventually adapted for the previously customized specific goals and selected responsibilities from groups of alternative or optional variants.

• Otherwise, if a set of similar specific goals were identified during the goal modeling task, thus reusing partially a goal model, then the set of role models already available in the corresponding domain model associated with the similar specific goal will be reused and eventually adapted, considering selected responsibilities from groups of alternative or optional variants.

• Otherwise, if the goal model was constructed from scratch, then the set of role models will be also constructed from scratch, one for each specific goal. Please note that in this task, reuse is implicitly supported by the semantic relationships, that associates a specific goal with a role model.

The role interaction modeling task aims at identifying how external and internal entities should cooperate to achieve a specific goal. For that, responsibilities of roles are analyzed along with their required and produced knowledge specified in a role model. A set of role interaction models is reused in this modeling task, one for each specific goal, or having it one or more variation points, one role interaction model for each variant, specifying the interactions between roles and external entities needed to achieve a specific goal. The interactions are numbered according to their sequencing. Similar rules to the ones of the role modeling task apply for the reuse activities performed during this modeling task. For the construction of the user interface prototype of the specific application,the generic interfaces associated to a reused external entity are selected and customized according the specific goal with which it is related.

3 RELATED WORK

Some approaches for agent development, like GAIA (Zambonelli et al., 2003), PASSI (Cossentino et al., 2004) and TROPOS (Bresciani et al., 2004), have been already developed to increase the productivity of the software development process, the reusability of generated products, and the effectiveness of project management.

GAIA is a methodology based in human organization concepts. It supports the analysis and design phases for multi-agent system development. Tropos is an agent-oriented software development methodology supporting the complete multi-agent development process. It is based on the i* organizational modeling framework. PASSI (a Process for Agent Societies Specification and Implementation) is a process for multi-agent development integrating concepts from object-oriented software engineering and artificial intelligence approaches. It allows the development of multi-agents systems for special purposes as mobiles and robotics agents and uses an UML-based notation.

Some characteristics of GAIA, PASSI, TROPOS and MADAE-Pro are following described. All the approaches propose an iterative life cycle, where a software product goes through several refinements during the development process. With the exception of GAIA, in all other approaches the life cycle is also incremental, where a software product is represented in several models to facilitate its understanding.

For the supported development phases, all these approaches cover analysis and design while PASSI, TROPOS and MADAE-Pro also support the implementation phase. To our knowledge, only MADAE-Pro allows the development of families of systems. For the available development tools, PASSI is supported by PTK, a Rational Rose plug-in allowing modeling in AUML and code generation. The application of TROPOS is assisted by the TAOM-Tool (Bresciani et al., 2004), an Eclipse plug-in allowing system modeling with the i* framework. The MADAE-Pro process is supported by the ONTORMAS tool that allows the modeling and storage of individual applications and families of multi-agent applications as instances of ontologies. GAIA does not have a tool support yet. Finally, for reuse activities, GAIA and TROPOS allow the reuse of models and code in an informal way. PASSI permits the reuse of source code from class and activity diagrams. MADAE-Pro process allows reuse of both models and source code of software products.

Two main features distinguish MADAE-Pro from other existing approaches. First, it provides support for reuse in multi-agent software development, through the integration of the concepts of Domain Engineering and Application Engineering. Second, it is a knowledge-based process where models of agents and frameworks are represented as instances of the ONTORMAS knowledge base. Thus, concepts are semantically related allowing effective searches and inferences thus facilitating the understanding and reuse of the

models during the development of specific applications in a domain. Also, the ontology-driven models of MADAE-Pro can be easily documented, adapted and integrated.

4 CONCLUSIONS AND FURTHER WORK

This work described MADAE-Pro, a process for Multi-agent Domain and Application Engineering, emphasizing the description of its domain analysis and application requirements engineering phases and showing how software artifacts produced on the first phase are reused in the last one.

The software artifacts produced by MADAE-Pro are represented as instances of the ONTORMAS tool, which serves as a repository of the MADAE-Pro reusable software artifacts and is a knowledge-based tool supporting application development.

MADAE-Pro has been evaluated with several case studies approaching both the development of application families (Mariano, Girardi, Leite, Drumond and Maranhão, 2008) and specific applications (Newton and Girardi, 2007). It makes part of a project for the improvement of multi-agent development techniques, methodologies and tools. With the knowledge base provided by ONTORMAS, an expert system is being developed, aiming at automating various tasks of both MADEM and MAAEM, thus allowing fast application development and partial code generation.

MADAE-Pro currently supports compositional reuse, based on the selection, adaptation and composition of software artifacts. A generative approach for reuse has been already explored with the specification of the GENMADEM methodology and the ONTOGENMADEM tool (Jansen and Girardi, 2006). ONTOGENMADEM provides support for the creation of Domain Specific Languages to be used on the generation of a family of applications in a domain. Further work will extend ONTORMAS for supporting ONTOGENMADEM allowing generative reuse in Multi-agent Applications Engineering.

REFERENCES

Bellifemine, F., Caire, G., Poggi, A., Rimassa, G., 2003. JADE A White Paper. Exp v. 3 n. 3, Sept., http://jade.tilab.com/.

Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., and Perini, A., 2004. “TROPOS: An Agent-Oriented Software Development Methodology”, In: Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers Volume 8, Issue 3, pp. 203-236.

Cossentino, M., Sabatucci, L., Sorace, S. and Chella, A., 2004. Patterns reuse in the PASSI methodology. In: Proceedings of the Fourth International Workshop Engineering Societies in the Agents World (ESAW'03), pp. 29-31, Imperial College London, UK.

Czarnecki, K., Eisenecker, U. W., 2000. Generative Programming: Methods, Tools, and Applications. ACM Press/Addison-Wesley Publishing Co., New York, NY.

Dileo, J., Jacobs, T. and Deloach, S., 2002. Integrating Ontologies into Multi-Agent Systems Engineering. Proceedings of 4th International Bi-Conference Workshop on Agent Oriented Information Systems (AOIS 2002), pp. 15-16, Bologna (Italy), July.

Leite, A. Girardi, R. Cavalcante, U.,2008. MAAEM: A Multi-agent Application Engineering Methodology. In: The 20th International Conference on Software Engineering and Knowledge Engineering, Redwood City. Proceedings of the 20th International Conference on Software Engineering and Knowledge Engineering. Boston: Knowledge Systems Institute.

Girardi, R.,1992. Application Engineering: Putting Reuse to Work. In: Dennis Tsichritzis (ed.). (Org.). Object Frameworks. Geneve: CUI, v. I, p. 137-149, Université de Geneve.

Jansen, M., Girardi, R., 2006. GENMADEM: A Methodology for Generative Multi-agent Domain Engineering. In: The 9th International Conference on Software Reuse, 2006, Torino. Proceedings of the 9th International Conference on Software Reuse, Lecture Notes in Computer Science (LNCS), v. 4039, p. 399-402. Berlin: Springer-Verlag.

Mariano, R., Girardi, R., Leite, A., Drumond, L. Maranhão, D., 2008. A Case Study on Domain Analysis of Semantic Web Multi-agent Recommender Systems. In: Proceedings 3th International Conference on Software and Data Technologies, p. 160-167, Porto. Portugal.

Newton, E., Girardi, R., 2007. PROPOST: A knowledge-based tool for supporting Project Portfolio Management. In: International Conference on Systems Engineering and Modeling - ICSEM'07, p. 98-103, Haifa. Proceedings of ICSEM'07.

Zambonelli, F., Jennings, N., Wooldridge, M. 2003 Developing multiagent systems: The Gaia methodology. ACM Transactions on Software Engineering and Methodology, pp. 417-470.