Istanbul Kültür University

1st International Workshop on Advanced Topics in Software Engineering (ATSEN)
Theme: Model-Driven Software Development

Model Driven Development of Multi-agent Systems

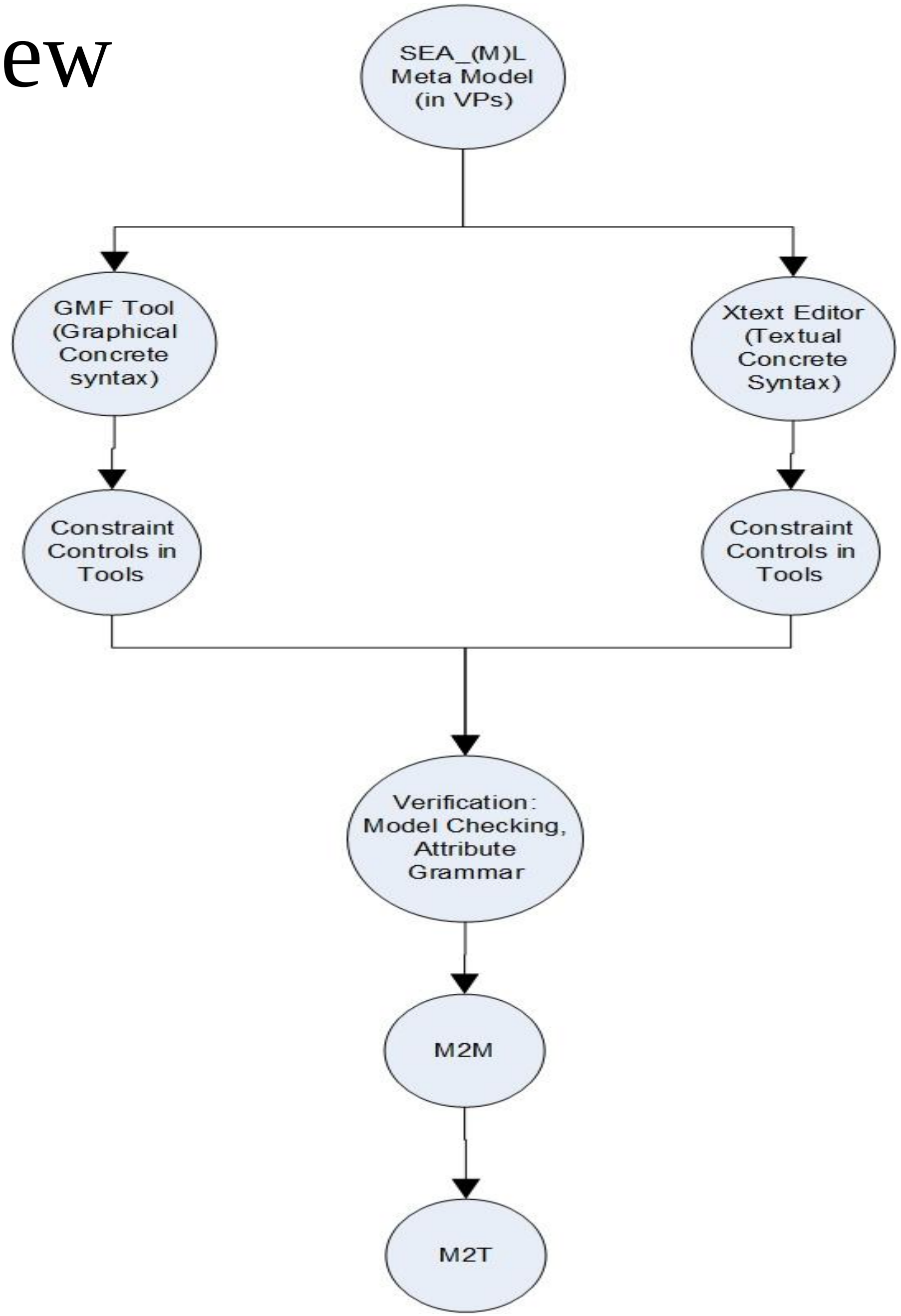
By: Moharram Challenger

- This study is financially supported by The Scientific and Technological Research Council of Turkey (TUBITAK) under grant 109E125 and Slovenian Research Agency (ARRS) under grant BI-TR/10-12-004.

Contents

- Introduction, the Problem and the Solution
- Project Overview
- PIM Level Abstract Syntax
- The Concrete Syntaxes:
  - TCS: SEA_L and its tooling
  - GCS: SEA_ML and its tooling
- The Operational Semantics
  - M2M Transformations to PSM level
  - M2T Transformations to Code level
- Case Studies and Evaluation

Project Overview



Introduction

Introduction - MAS

- **Software agents**: Autonomous software components capable of acting on behalf of their human users in order to perform a group of defined tasks
- **MAS (Multi Agent System)**:

  Systems in which many intelligent software agents interact with each other, either cooperative or selfish [Sycara, 1998]
- **AOSE** (Agent Oriented Software Engineering) deals with:
  - Agent Communication Lang, Interaction Protocols,
  - Architectures: e.g. BDI, Practical Reasoning, Deductive reasoning, Reactive agents
  - Development methodology: e.g. Tropos, Gaia, INGENIAS
  - Platforms: JADE, JADEX, SEAGENT, JACK

Introduction – Semantic Web Enabled MAS

- Semantic Web: allows machines to understand the meaning or semantics of info on the Web
- The agents would be able to perform tasks automatically and locate related information on behalf of the user
- SW let web-pages to be interpreted with ontologies
- This Interpretation can be done via Agents
- Thus, SW enabled MASs are required

Introduction – Problem

- MASs are complex, even more complex with Semantic Web
- Need more abstraction and a Meta-modeling can be a solution
- MDD changes focus of development from code to model
- DSML for MASs provides platform independent modeling
- DSML allows end-users & domain experts to specify a problem using visual abstractions close to domain
  - In this way, they reduce the complexity

ECOOP Conference, Maribor, Slovenia, 2010

Towards a DSML for Semantic Web enabled Multi-agent Systems

Geylani Kardas
International Computer Institute
Ege University
35100 Bornova, Izmir, TURKEY
Tel: +90-232-3423232-103
geylani.kardas@ege.edu.tr

Zekai Demirezen
Department of Computer and Information Sciences
University of Alabama at Birmingham
35294-1170 Birmingham, AL, USA
zekzek@uab.edu

Moharram Challenger
International Computer Institute
Ege University
35100 Bornova, Izmir, TURKEY
Tel: +90-232-3423232-119
challenger@engineer.com

ABSTRACT

Software agents are considered as autonomous software components which are capable of acting to meet its design objectives. To perform their tasks and interact with each other, agents constitute systems called Multi-agent systems (MAS). Although agent researchers have a great effort in MAS metamodeling and model-driven MAS development, a significant deficiency exists in current studies when we consider providing a complete Domain Specific Modeling Language (DSML) for MASs. We believe that a DSML increases the descriptive power of a MAS metamodel, defines the system semantics and hence supports a more fruitful methodology for the development of MASs especially working on the new challenging environments such as the Semantic Web. In this paper, we introduce a new DSML for MASs with its abstract syntax, the textual concrete syntax and the interpreter mechanism. The practical use of the DSML is illustrated with a case study which considers the modeling of a multi-agent based e-barter system.

Categories and Subject Descriptors

D.3.1 [**Programming Languages**]: Formal Definitions and Theory – *semantics, syntax* I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Multiagent systems*

General Terms

Algorithms, Design, Languages.

Keywords

Domain Specific Modeling Language, Metamodel, Model-Driven Engineering, Multi-agent System, Semantic Web.

1. INTRODUCTION

Software agents are considered as autonomous software components which are capable of acting on behalf of their human users in order to perform a group of defined tasks. Many intelligent software agents interact with each other in a system that we call Multi-agent Systems (MASs). The implementation of these autonomous, responsive and proactive software systems is naturally a complex task. In addition, internal agent behavior model and interaction within the agent organizations become even more complex and hard to implement when new requirements and interactions for new agent environments such as the Semantic Web [1] are considered. To work in a higher abstraction level is of critical importance for the development of MASs since it is almost impossible to observe code level details of MASs due to their internal complexity, distributedness and openness. Within this context, Model-Driven Engineering (MDE) [2] may provide an infrastructure that simplifies the development of MASs.

MDE has been shown to increase productivity and reduce development costs [3]. It provides higher levels of abstraction to allow such users to focus on the problem, rather than the specific solution [2]. In particular, domain-specific modeling (DSM) is a modeling approach that provides languages, called Domain Specific Modeling Languages (DSMLs), that fit the domain of an end-user by offering intentions, abstractions, and visualizations for domain concepts [4]. Therefore, the part of the success of MDE is also dependent on the descriptive power of DSMLs [5].

Although there exists a great effort of agent researchers in MAS metamodeling (e.g., [6, 7]) and model-driven MAS development (e.g., [8, 9]), a significant deficiency exists in current studies when we consider providing a complete DSML for MASs. In our previous work, we also defined an agent metamodel [10] and have recently presented a complete MDE process [11] for rapid implementation of MASs on various agent software platforms. The proposed metamodel especially supports the Semantic Web constructs and their interactions with the traditional agent system components to provide MDE of the Semantic Web enabled MASs. However, similar to the above referenced studies, both formal specification and descriptive power of the given metamodel are not sufficient enough for the definition of the system semantics and the verification of the MAS models conforming to our proposed metamodel. This is a fundamental requirement especially when we consider dynamic MAS models describing agent behaviors and interactions both with the other agents and the semantic web services. We believe that the definition of a new DSML for such MASs would remove above discussed shortages originating from the existing metamodel and enable agent developers to use a more fruitful MDE methodology for the development of MASs especially working on the new Semantic Web environment. Hence, in this paper we present the initial results of our ongoing study on defining a new DSML that can be used during the model-driven MAS development.

ECOOP '2010 Maribor, Slovenia EU

Abstract Syntax

Abstract Syntax - Overview

- A meta model inspired from
  - FIPA ACSM from IEEE
  - UML components
  - Kardas et al. 2009 study
  - OMG’s ODM (Ontology Definition Metamodel)
  - BDI agent’s internal meta-elements

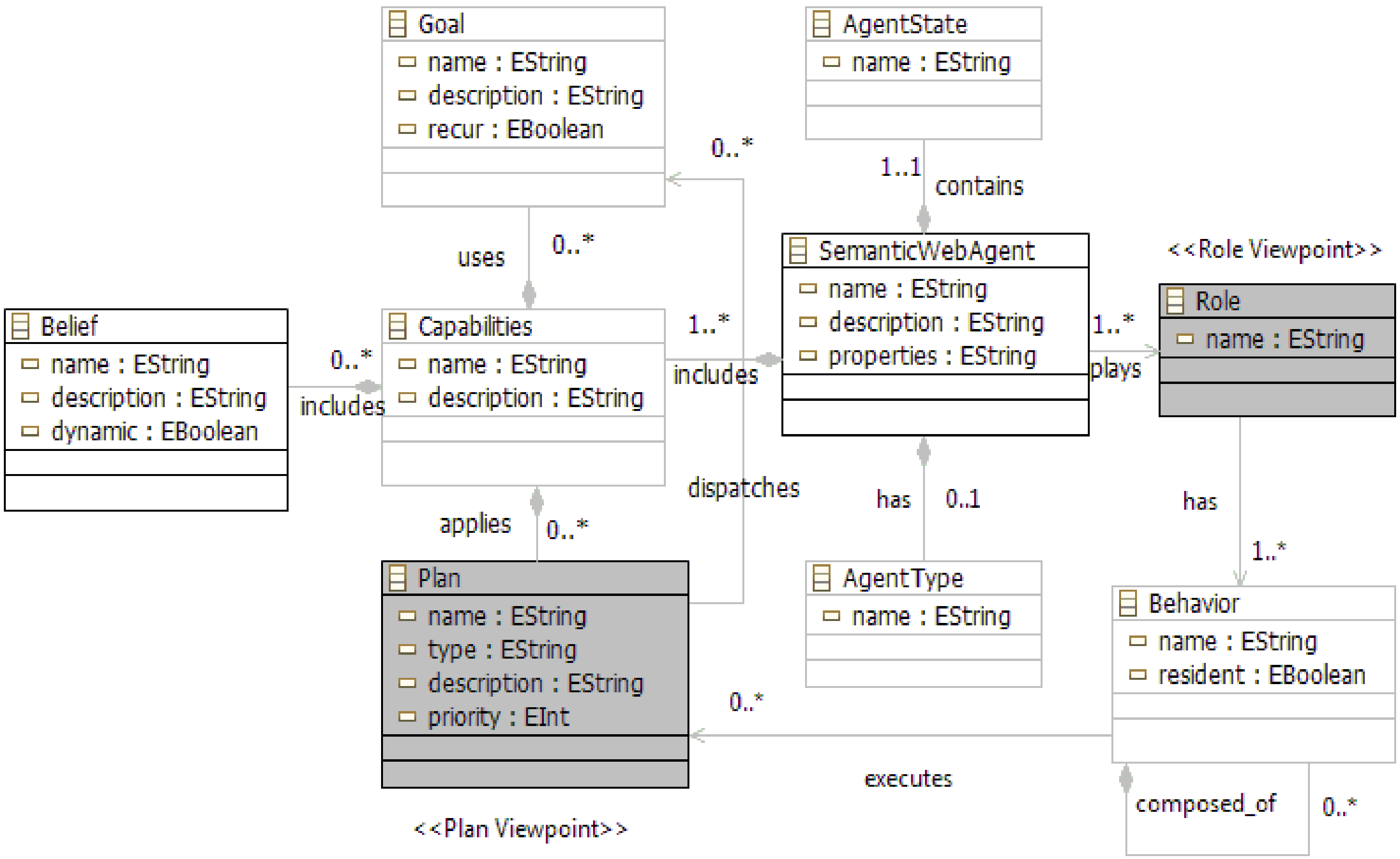
Abstract Syntax

The abstract syntax is introduced in 8 viewpoints:

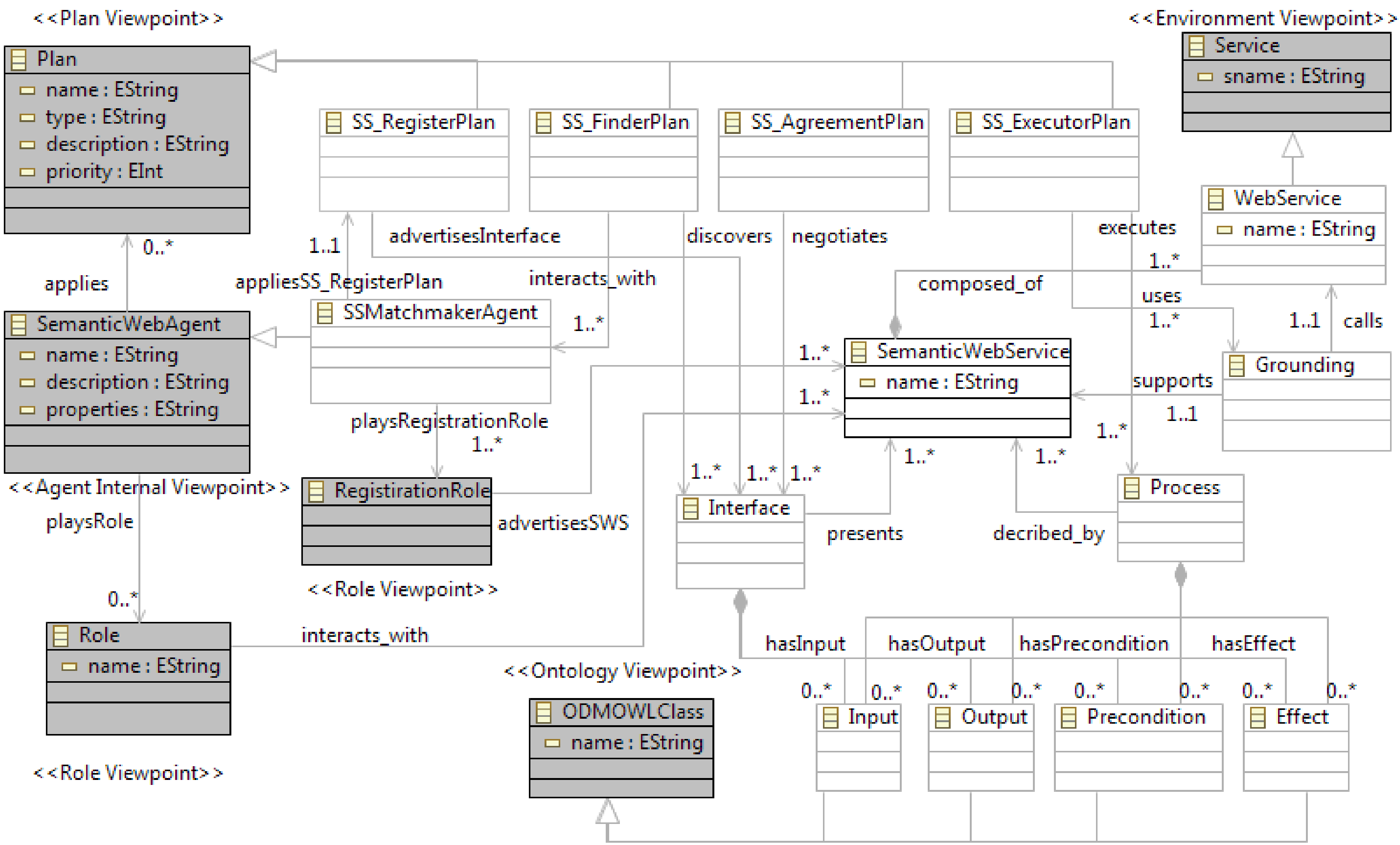
1. Agent Internal
2. Plan
3. Role
4. Interaction
5. Environment
6. MAS
7. Ontology
8. Agent-SWS Interaction

- The abstract syntax is provided in EMF Ecore format

Agent Internal Viewpoint



Agent-SWS viewpoint



Lecture notes in Business Processing, Vol. 83, pp. 177-186, 2011 (CAiSE Workshop, London)

A Domain Specific Metamodel for Semantic Web Enabled Multi-Agent Systems

Moharram Challenger, Sinem Getir, Sebla Demirkol, and Geylani Kardas

International Computer Institute, Ege University, 35100, Bornova, Izmir, Turkey
moharram.challenger@mail.ege.edu.tr,
{sinem.getir,sebla.demirkol,geylani.kardas}@ege.edu.tr

Abstract. Autonomous, responsive and proactive nature of agents makes development of agent-based software systems more complex than other software systems. A Domain Specific Modeling Language (DSML) may provide the required abstraction and hence support a more fruitful methodology for the development of MASs especially working on the new challenging environments such as the Semantic Web. In this paper, we introduce a domain specific metamodel for MASs working on the Semantic Web. This new metamodel paves the way for definition of an abstract syntax and a concrete syntax for a future DSML of agent systems. Achieved DSML syntax is supported with a graphical modeling toolkit.

Keywords: metamodel, domain specific modeling language, multi-agent system, semantic web.

1 Introduction

Development of intelligent software agents keeps its emphasis on both artificial intelligence and software engineering research areas. In its widely-accepted definition, an agent is an encapsulated computer system (mostly a software system) situated in some environment, and that is capable of flexible autonomous action in this environment in order to meet its design objectives [1]. These autonomous, reactive and proactive agents have also social ability and interact with other agents and humans in order to complete their own problem solving. They may also behave in a cooperative manner and collaborate with other agents to solve common problems. To perform their tasks and interact with each other, intelligent agents constitute systems called Multi-agent systems (MAS).

Considering abovementioned characteristics, the implementation of agent systems is naturally a complex task. In addition, internal agent behaviour model and interaction within the agent organizations become even more complex and hard to implement when new requirements and interactions for new agent environments such as the Semantic Web [2] are taken into account. Semantic Web brought a new vision into agent research. This new generation Web aims to improve World Wide Web (WWW) such that web page contents are interpreted with ontologies. It is apparent that the interpretation in question will be realized by autonomous computational entities –so

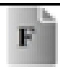
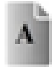
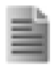
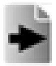
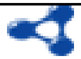
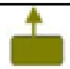
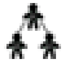
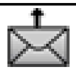
C. Salinesi and O. Pastor (Eds.): CAiSE 2011 Workshops, LNBIP 83, pp. 177–186, 2011.

The Concrete Syntaxes (SEA_L and SEA_ML), Their Constraints, and Tools

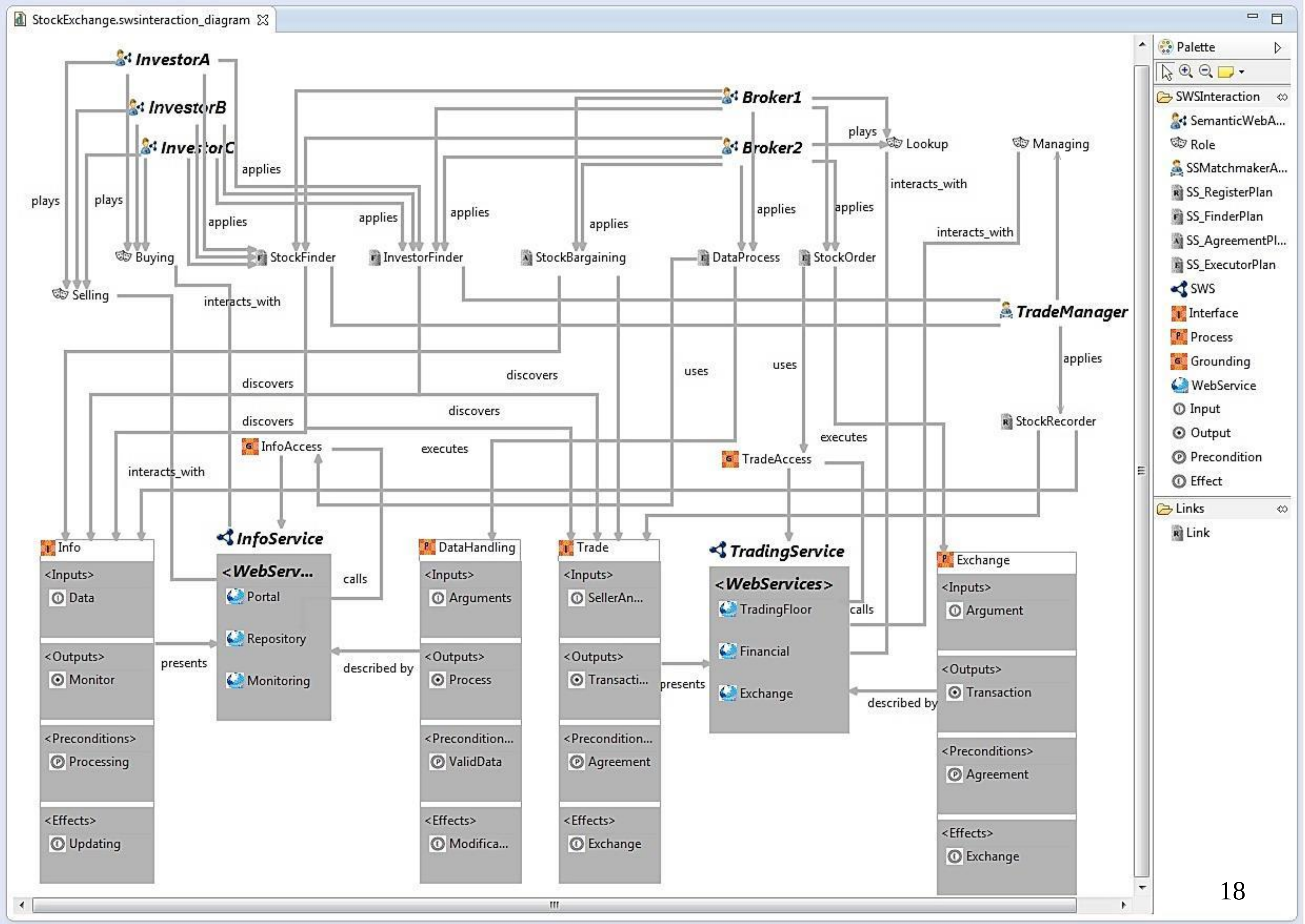
Graphical Concrete Syntax

- Concrete syntax: The mapping between meta elements and graphical and/or textual notations
  - In this study both of the notations are provided for the domain
- The graphical concrete syntax is developed in GMF
- Its constraints is defined in OCL
- The integrated tooling is provided in Eclipse framework

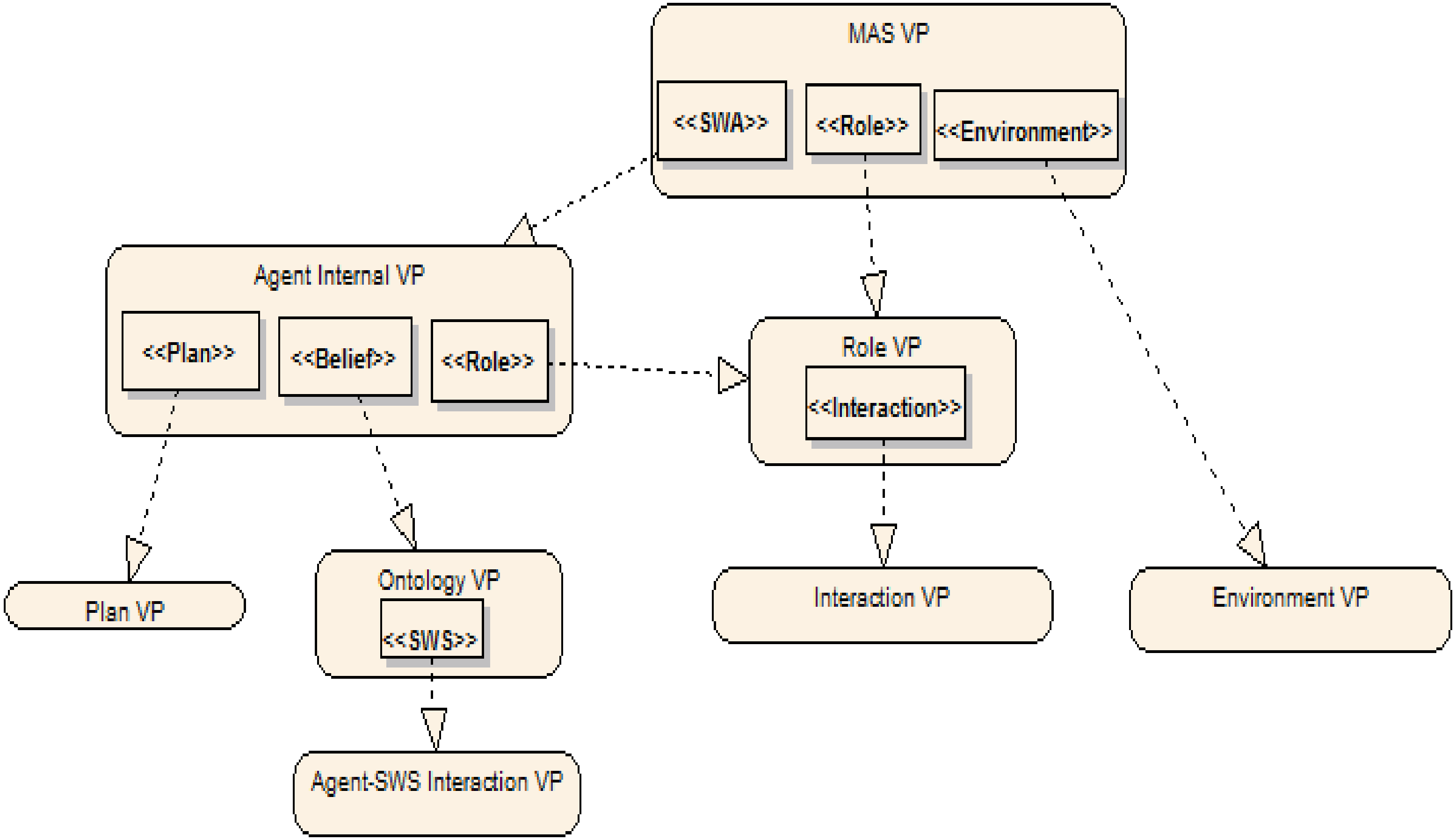
Some of the notations provided for GCS

Concept	Notation	Concept	Notation
Belief		Semantic Service yMatchmaker Agent	
Fact		Semantic Service–Register Plan	
Goal		Semantic Service-Finder Plan	
Capabilities		Semantic Service-Agreement Plan	
Plan		Semantic Service–Execution Plan	
Behaviour		Registration Role	
Action		Service Ontology	
Agent State		Semantic Web Service	
Agent Type		Scenario	
Role		Domain Role	
Task		Semantic Web Agent	
Web service		Semantic Web Organization	
Resource		Role Ontology	
Environment		Protocol	
Permission Table		Message	
Message Type		Interaction/Contract net	

An instance model in the SEA_ML tool



System integration and transitions between SEA_ML viewpoints



SPLASH Workshop, Oregon USA, 2011

The GMF-based Syntax Tool of a DSML for the Semantic Web enabled Multi-Agent Systems

Sinem Getir
Int'l Computer Institute
EGE Univ., Izmir, Turkey
Tel: +90 232 311 3234
sinem.getir@ege.edu.tr

Sebla Demirkol
Int'l Computer Institute
EGE Univ., Izmir, Turkey
Tel: +90 232 311 3233
sebla.demirkol@ege.edu.tr

Moharram Challenger
Int'l Computer Institute
EGE Univ., Izmir, Turkey
Tel: +90 232 311 3252
moharram.challenger@mail.ege.edu.tr

Geylani Kardas
Int'l Computer Institute
EGE Univ., Izmir, Turkey
Tel: +90 232 311 3223
geylani.kardas@ege.edu.tr

Categories and Subject Descriptors

D.3.1 [**Formal Definitions and Theory**]: *Formal Definitions and Theory – Semantic, Syntax* D.3.3 [**Languages Constructs and Features**]: *Constraints*. D.2.11 [**Software Architectures**]: *Domain-specific architectures*, D.2.13 [**Reusable Software**]: *Domain Engineering*, I.2.11 [**Distributed Artificial Intelligence**]: *Intelligent agents, Multi agent system*

General Terms

Design, Languages

Keywords

Domain Specific Modeling Languages, Metamodel, Multi Agent Systems, Semantic Web.

1. INTRODUCTION

Internal complexity of agents such as being autonomous, reactive and proactive makes development of agent-based software systems complex. On the other hand, one of the most appropriate environments for Multi Agent Systems (MAS) is Semantic Web. Internal agent behavior model and interaction within the agent organizations become even more complex to implement when new requirements and interactions for environments such as the Semantic Web. A Domain Specific Modeling Language (DSML) can provide the required abstraction and hence support a more fruitful methodology for the development of MAS especially working on the new challenging environments such as the Semantic Web. The new abstract and concrete syntaxes pave the way for developing a future DSML of agent systems, called Semantic web Enabled Agent Modeling Language (SEA_ML) [1].

SEA_ML is a DSML which supports the development of Multi-agent Systems working on the Semantic Web. In our vision, the "Semantic Web enabled MAS" means that software agents are planned to collect Web content from diverse sources, process the information and exchange the results on behalf of their human users. Autonomous agents can also evaluate semantic data within these MASs and collaborate with semantically defined entities such as Semantic Web services by using content languages. We call the software agents with these capabilities as Semantic Web Agents.

In [1], we discuss how domain specific engineering can provide easy and rapid construction of Semantic Web enabled MASs and introduce SEA_ML. Similar to other DSMLs; SEA_ML should provide complete definitions for its abstract syntax, concrete syntax and formal semantics. We discuss the abstract syntax of SEA_ML in [2]. A meta-model that describes the meta-entities and their relationships for a domain can naturally provide a base for the definition of such an abstract syntax. Our meta-model for the Semantic Web enabled MASs considers various aspects of MAS development (e.g., role, protocol and organizational) and provides both internal modeling of a software agent and interaction of agents and Semantic Web services within the environment. In this demo paper, we present the graphical concrete syntax tool for SEA_ML.

2. GRAPHICAL SYTAX TOOL OF SEA_ML

This section introduces the developed GMF-based [3] graphical tool for SEA_ML. Capabilities, constraints, rules and facilities of the tool are discussed in the first subsection. Afterwards, we illustrate the tool properties within an e-barter case study showing how constraints can be provided.

2.1 The Tool Overview

The tool we developed supplies a graphical editor which provides to choose model elements from the palette. These elements can be filled with the labels and properties in order to complete the full features of model elements with optional and different notations and colors of the graphical facilities. The model also defines the restrictions and main elements of the modeled MASs and provides an overview of the agents of a complex instance model for agent developers.

The Ecore [3] models are created in Eclipse. After setting the graphical notations for abstract syntax meta elements, we use Eclipse GMF to tie notations to the domain concepts. Provided GMF tools can be useful for the modeling of MASs and getting an overview of the agents of a complex instance models for agent developers. Some of the graphical notations for Agent internal and Agent-SWS Interaction viewpoints are illustrated in Table 1. Odd columns define the names of the meta elements in abstract syntax and even columns mark notations, or icons in the tool.

Textual Concrete Syntax

- The textual concrete syntax is developed in Xtext
  - The abstract syntax is converted to EBNF rules
- The constraints are implemented using Xtext packages
  - Formatting package
  - Validation package
  - Scoping package
- Using these capabilities, the new DSL possesses static semantics of the MAS domain

Sample Program in SEA_L

Ebarter.agent

```
AgentInternalViewPoint e_barter{
    SemanticWebAgent barterManager
        "Barter Manager Agent"   // Agent Description
        "Properties"             // Agent Properties
        "Active"                 // Agent State
        "CustomerAgent";         // Agent Type
    Capabilities barterCap
        "Barter Manager Capability";
    Role barterRole;
    Goal bestMatching
        "Doing best matching" 1; // Recur = 1
    Belief barterKnowledge
        "system facts" 2;         // Dynamic = 2
    Plan financialPlan
        "Cyclic Plan" 1;          // Priority = 1
    barterManager{
        includes barterCap;
        plays barterRole;
    }
    barterCap{
        appliesPlan financialPlan;
        includesBelief barterKnowledge;
        usesGoal bestMatching;
    }
    barterKnowledge{
        precondition bestMatching;
    }
    bestMatching{
        postcondition barterKnowledge;
        realized_by financialPlan;
    }
}
```

Textual Concrete Syntax

The steps we followed:

- First, Ecore model is converted to Xtext EBNF grammar rules
- Second, the generated EBNF is modified to desire form
- Third, the textual representations are selected for meta element

- This tool can validate the instance model based on its grammar

FedCSIS Conference, Wroclaw, Poland, 2012

Proceedings of the Federated Conference on Computer Science and Information Systems pp. 1373–1380 ISBN 978-83-60810-48-4

SEA_L: A Domain-specific Language for Semantic Web enabled Multi-agent Systems

Sebla Demirkol[1] sebla.demirkol@ege.edu.tr
Moharram Challenger[1] moharram.challenger@mail.ege.edu.tr
Sinem Getir[1] sinem.getir@ege.edu.tr
Tomaž Kosar[2] tomaz.kosar@uni-mb.si
Geylani Kardas[1] geylani.kardas@ege.edu.tr
Marjan Mernik[2] marjan.mernik@uni-mb.si

[1]International Computer Institute, EGE University, Bornova, 35100 Izmir, Turkey
[2]Faculty of Electrical Engineering and Computer Science, University of Maribor, Maribor, Slovenia

Abstract— Autonomous, reactive and proactive features of software agents make development of agent-based software systems complex. A Domain-specific Language (DSL) can provide the required abstraction and hence support a more fruitful methodology for the development of Multi-agent Systems (MASs) especially working on the new challenging environments such as the Semantic Web. Based on our previously introduced domain-specific metamodel, in this paper we propose a textual concrete syntax of a DSL for MASs working on the Semantic Web and show how the specifications of this DSL can be utilized during the code generation of exact MASs. The new DSL is called Semantic web Enabled Agent Language (SEA_L). The syntax of SEA_L is supported with textual modeling toolkits developed with Xtext. The practical use of SEA_L is illustrated with a case study which considers the modeling of a multi-agent based e-barter system.

Keywords— Domain-specific Languages; Metamodel; Multi-agent Systems; Semantic Web

I. Introduction

SOFTWARE agents [1] are autonomous software components which are able to act on behalf of their users in order to do a group of defined tasks. Many intelligent software agents interact with each other in a system that is called Multi-agent System (MAS). Their interactions can be either cooperative or selfish [2]. Software agents and MASs are recognized as both useful abstractions and effective technologies for the modeling and building of complex distributed systems. The implementation of these autonomous, responsive and proactive systems is naturally a complex task.

Additionally, Semantic Web improves World Wide Web such that web page contents can be interpreted with ontologies [3]. Therefore, this new generation web helps machines to understand web content. It is apparent that the interpretation in question will be realized by autonomous computational entities (i.e. agents) to handle the semantic content on behalf of their users. Surely, Semantic Web environment has specific architectural entities and a different semantic which must be considered to model a MAS within this environment. Thus, Semantic Web evolution brought a new vision into agent research. Software agents are planned to collect Web content from diverse sources, process the information and exchange the results. Autonomous agents can also evaluate semantic data and collaborate with semantically defined entities of the Semantic Web such as semantic web services by using content languages. However, considering agent interactions with Semantic Web elements adds more complexity for designing and implementing those systems.

On the other hand, Model Driven Development (MDD) is one of the important software development approaches, moving software development from code to models [4] which increases productivity [5] and reduces development costs [6]. Design and implementation of a MAS may become more complex when new requirements and interactions for new agent environments such as Semantic Web are considered. MDD can provide an infrastructure that simplifies the development of such MASs. To work in a higher abstraction level is of critical importance for the development of MASs since it is almost impossible to observe code level details of the MASs due to their internal complexity, distributedness and openness. Hence, such MDD application can increase the abstraction level in MAS development. MDD uses different approaches to realize its goals. One of these methods is Domain Specific Language (DSL) development [7, 8, 9, 10, 11]. DSLs are languages which comprise a domain's concepts and terminologies to supply the requirements of the domain. A DSL allows end-user programmers (domain experts) to describe the essence of a problem with abstractions related to a domain specific problem space.

A domain specific metamodel for semantic web enabled MASs is discussed in [12]. Based on this metamodel, in this paper, we present the textual concrete syntax of a DSL and discuss transformations required for code generation from the specifications of this DSL. We call this new DSL as Semantic web Enabled Agent Language (SEA_L).

The rest of this paper is organized as follows: Related work is given in Section 2. The abstract syntax and the textual concrete syntax of SEA_L are discussed in Sections 3 and 4 respectively. In section 5, the code generation mechanism for new DSL is illustrated. Section 6 includes a case study on the development of a MAS by using SEA_L. Finally, Section 7 concludes the paper and states the future work.

This study is funded as a bilateral project by The Scientific and Technological Research Council of Turkey (TUBITAK) under grant 109E125 and Slovenian Research Agency (ARRS) under grant BI-TR/10-12-004.

The Operational Semantics

Transformations

- The operational semantics is realized using transformations
- The artifacts are automatically generated using two phase transformations:
  - Model to Model transformations (M2M)
  - Model to Text transformations (M2T)
- M2M transformations for PIM to PSM conversion
  - From SEA_ML to Agent target languages: JADEX and JACK
  - From SEA_ML to Semantic Web tech.: OWL-S and WSMO
- M2T transformations for PSM to Code conversion
  - From JADEX and JACK models to their target language
  - From OWL-S and WSMO models to their documents

Transformations

- The M2M transformations are fulfilled with ATL
- The M2T transformations are realized using:
  - Mofscript in SEA_ML and
  - Xpand in SEA_L

- All of the above mentioned tools are EMF compatible.

E.g. Code Generation for SEA_L

- The partial mapping table used for transformation between SEA_L, Jadex and OWL-S

SEA_L	Jadex	OWLS
SemanticWebAgent	Agent	
SSMatchmakerAgent	Agent	
Plan, Behavior	Plan	
Capabilities	Capability	
Goal	AchieveGoal	
Goal	QueryGoal	
Goal	PerformGoal	
SS_AgreementPlan	Plan	
SS_ExecutorPlan	Plan	
SS_FinderPlan	Plan	
SS_RegisterPlan	Plan	
SemanticWebService		Service
Interface		ServiceProfile
Process		ServiceModel
Grounding		ServiceGrounding
Input		Input
Output		Output
Precondition		Condition
Effect		ResultVar

ComSIS Journal, vol. 10, no 4, pp.1525-1556, 2013

DOI:10.2298/CSIS121105044D

A DSL for the Development of Software Agents working within a Semantic Web Environment

Sebla Demirkol[1], Moharram Challenger[1], Sinem Getir[1], Tomaž Kosar[2], Geylani Kardas[1*], and Marjan Mernik[2]

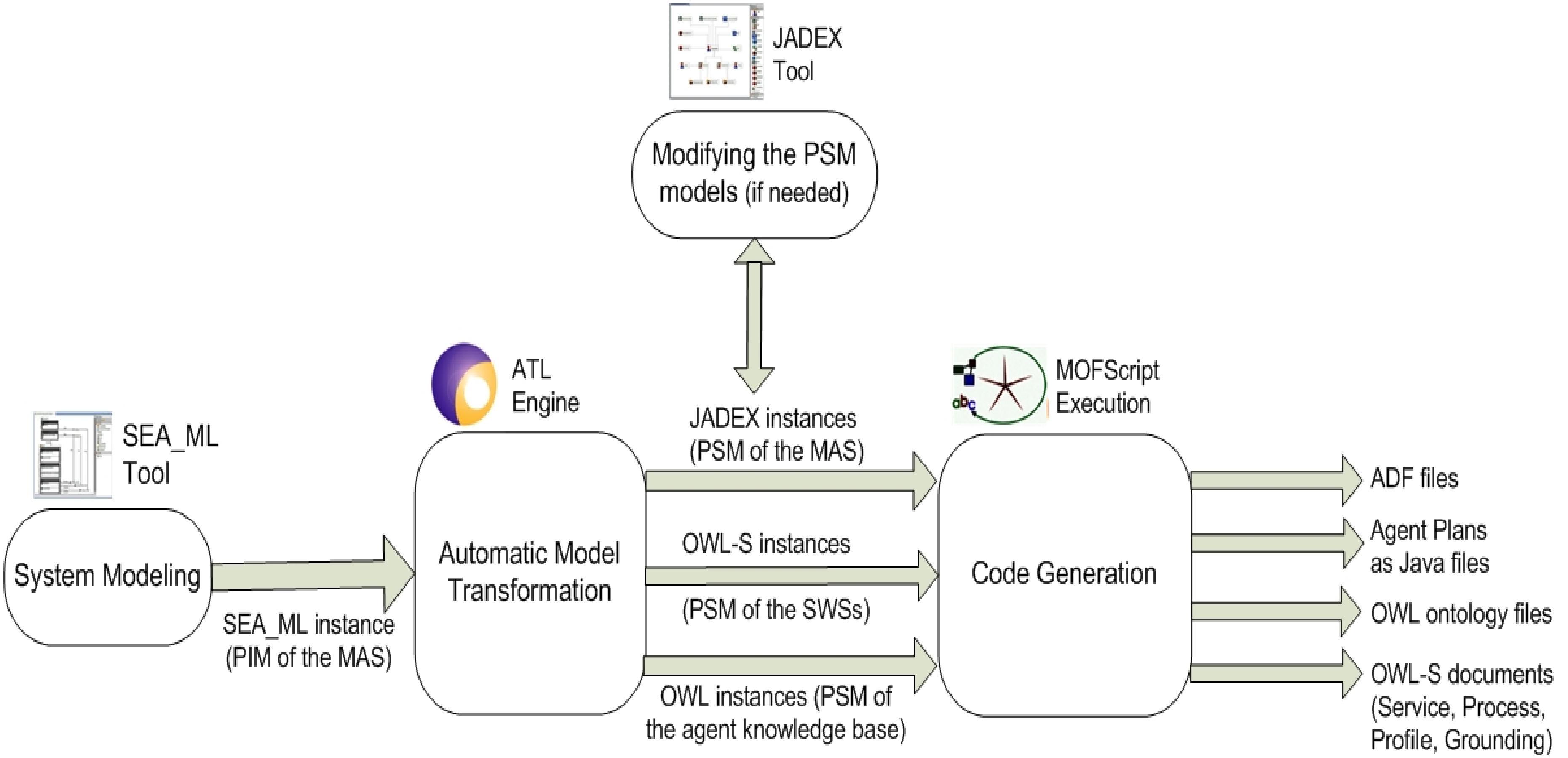
[1] International Computer Institute, Ege University, Bornova, 35100 Izmir, Turkey
sebla.demirkol@ege.edu.tr, moharram.challenger@mail.ege.edu.tr, sinem.getir@ege.edu.tr, geylani.kardas@ege.edu.tr

[2] Faculty of Electrical Engineering and Computer Science, University of Maribor, Smetanova 17, 2000 Maribor, Slovenia
tomaz.kosar@uni-mb.si, marjan.mernik@uni-mb.si

Abstract. Software agents became popular in the development of complex software systems, especially those requiring autonomous and proactive behavior. Agents interact with each other within a Multi-agent System (MAS), in order to perform certain defined tasks in a collaborative and/or selfish manner. However, the autonomous, proactive and interactive structure of MAS causes difficulties when developing such software systems. It is within this context, that the use of a Domain-specific Language (DSL) may support easier and quicker MAS development methodology. The impact of such DSL usage could be clearer when considering the development of MASs, especially those working on new challenging environments like the Semantic Web. Hence, this paper introduces a new DSL for Semantic Web enabled MASs. This new DSL is called Semantic web Enabled Agent Language (SEA_L). Both the SEA_L user-aspects and the way of implementing SEA_L are discussed in the paper. The practical use of SEA_L is also demonstrated using a case study which considers the modeling of a multi-agent based e-barter system. When considering the language implementation, we first discuss the syntax of SEA_L and we show how the specifications of SEA_L can be utilized during the code generation of real MAS implementations. The syntax of SEA_L is supported by textual modeling toolkits developed with Xtext. Code generation for the instance models are supplied with the Xpand tool.

Keywords: Domain-specific Language, Metamodel, Multi-agent

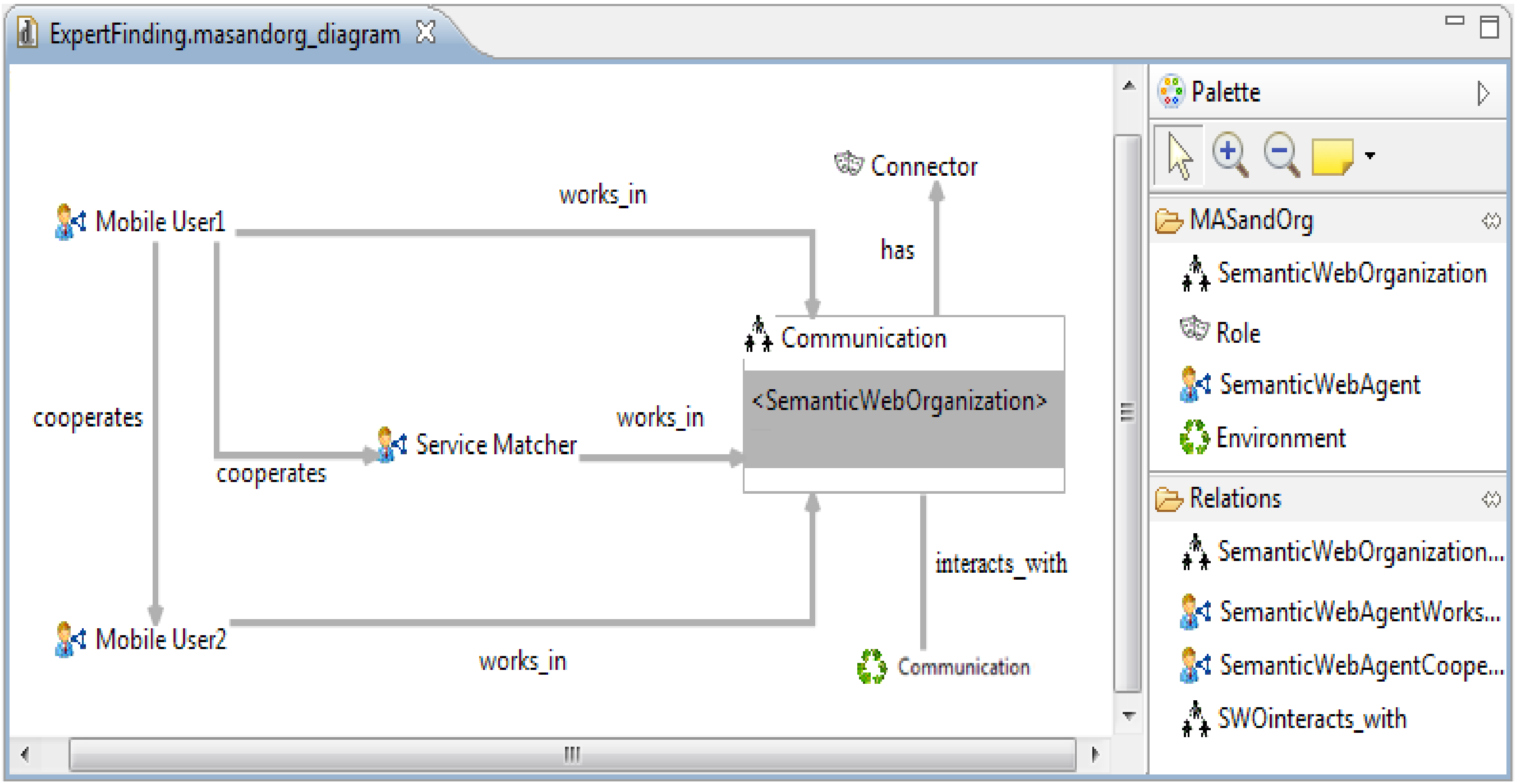
The use of SEA_ML



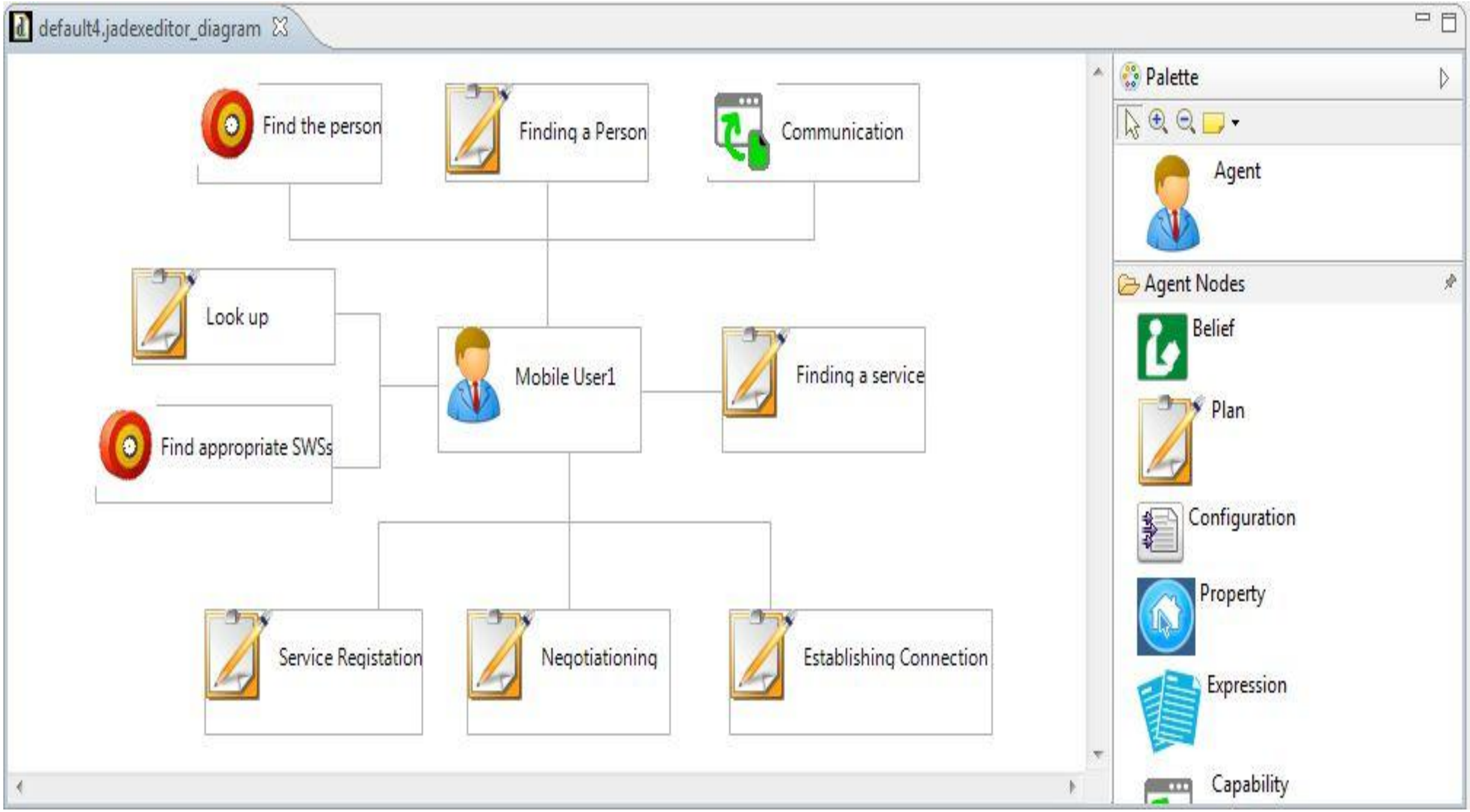
Case Studies

- E-Barter System (Published in INISTA 2012)
- Stock Exchange System (Published SPE Journal 2012)
- Conference Management System
- Expert Finding System
- Hotel Reservation System

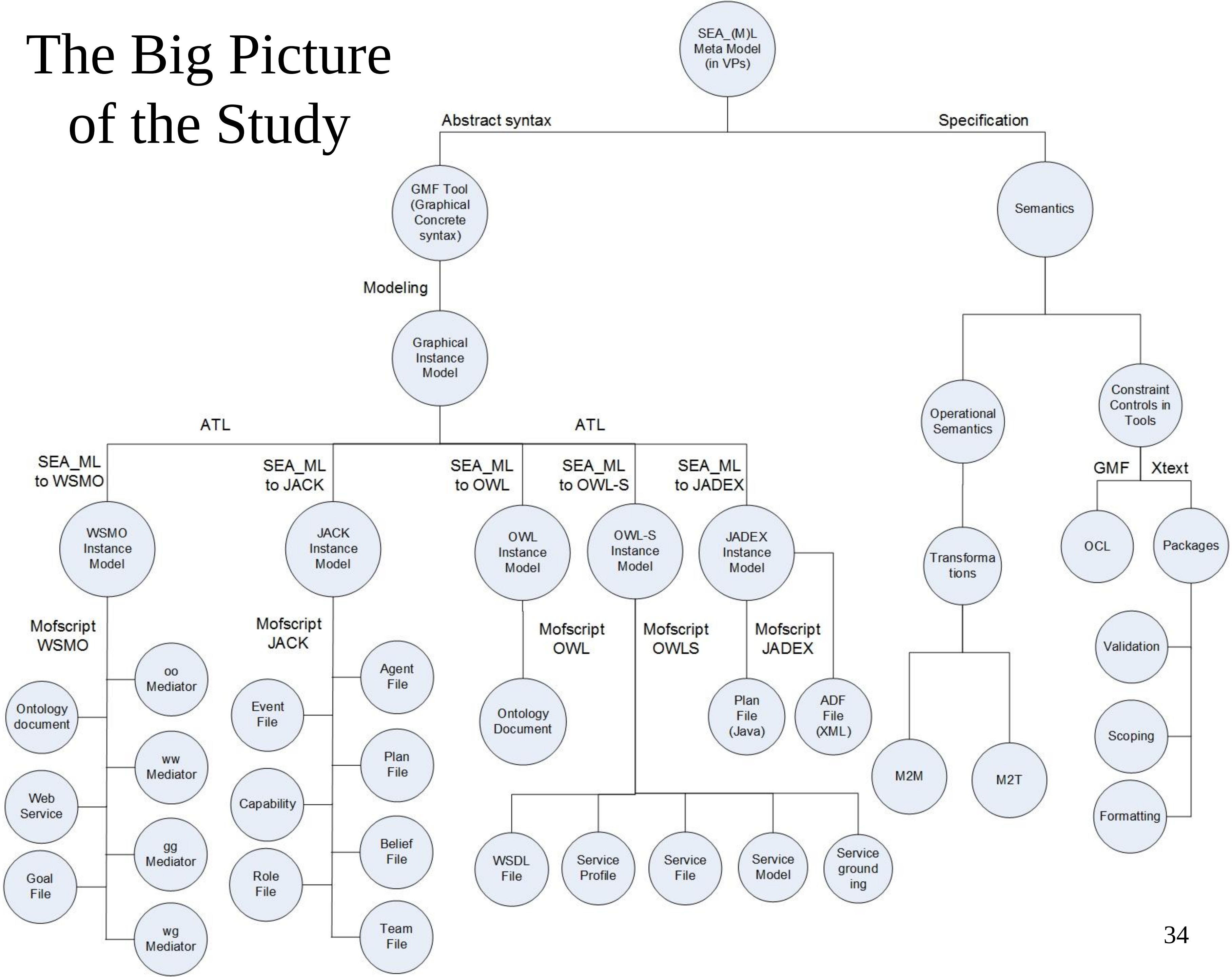
Case study – Expert Finding – PIM MAS VP



Case study – Expert Finder –PSM Agent VP



The Big Picture of the Study



Comparison the study with literature

	Abstract syntax definition	Concrete syntax definition	Viewpoint inclusion	M2M transformability	M2C transformability	Tool support
Bauer et al., 2001	++	+	AIV, IV, BV	-	-	+
Cervenka et al., 2005	++	+	AIV, MASV, IV, BV, EV	-	-	+
Kulesza et al., 2005	++	+	MASV, IV, RV, GV	-	+	+
Pavon et al., 2006	++	+	AIV, MASV, IV, GV, EV	-	+	+
Rougemaille et al., 2007	+	-	AIV, MASV, IV	+	-	-
Hahn, 2008*	++	+	AIV, MASV, OV, IV, RV, BV, EV	++	+	++
Hahn et al., 2008	++	+	AIV, MASV, OV, IV, RV, BV, EV, ASIV	+	+	++
Kardas et al., 2009a	++	-	-	++	+	++
Fuentes-Fernandez et al., 2010	+	+	AIV, IV, GV	-	+	+
Ciobanu and Juravle, 2012	+	+	-	-	++	+
Gascuena et al., 2012	+	+	AIV, RV, PV, GV	++	+	++
SEA_ML	++	++	AIV, MASV, IV, PV, RV, EV, ASIV	++	++	++

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

On the use of a domain-specific modeling language in the development of multiagent systems

Moharram Challenger [a], Sebla Demirkol [a], Sinem Getir [a], Marjan Mernik [b], Geylani Kardas [a,*], Tomaž Kosar [b]

[a] International Computer Institute, Ege University, Bornova, 35100 Izmir, Turkey
[b] Faculty of Electrical Engineering and Computer Science, University of Maribor, Smetanova 17, 2000 Maribor, Slovenia

ARTICLE INFO

Article history:
Received 3 June 2012
Received in revised form
8 November 2013
Accepted 19 November 2013
Available online 21 December 2013

Keywords:
Agent
Multiagent system
Model driven development
Domain-specific modeling language
Metamodel
Semantic web

ABSTRACT

The study of Multiagent Systems (MASs) focuses on those systems in which many intelligent agents interact with each other. The agents are considered to be autonomous entities which contain intelligence that serves for solving their selfish or common problems, and to achieve certain goals. However, the autonomous, responsive, and proactive natures of agents make the development of agent-based software systems more complex than other software systems. Furthermore, the design and implementation of a MAS may become even more complex and difficult to implement when considering new requirements and interactions for new agent environments like the Semantic Web. We believe that both domain-specific modeling and the use of a domain-specific modeling language (DSML) may provide the required abstraction, and hence support a more fruitful methodology for the development of MASs. In this paper, we first introduce a DSML for MASs called SEA_ML with both its syntax and semantics definitions and then show how the language and its graphical tools can be used during model-driven development of real MASs. In addition to the classical viewpoints of a MAS, the proposed DSML includes new viewpoints which specifically support the development of software agents working within the Semantic Web environment. The methodology proposed for the MAS development based on SEA_ML is also discussed including its example application on the development of an agent-based stock exchange system.

Evaluation

Evaluation – preparation

Groups	Number of members	Number of Ph.D candidates	Number of Master students	Type of Evaluation
A	5	1	4	Using SEA_ML
B	5	1	4	Manual

$$C_{TotalInit} = C_{Teaching} + C_{ReadingDocs} + C_{Demo}$$

$$C_{TotalDevelop} = C_{Analysis} + C_{Modeling/Design} + C_{Implementation} + C_{ErrorDetection} + C_{Maintainance}$$

Evaluation result

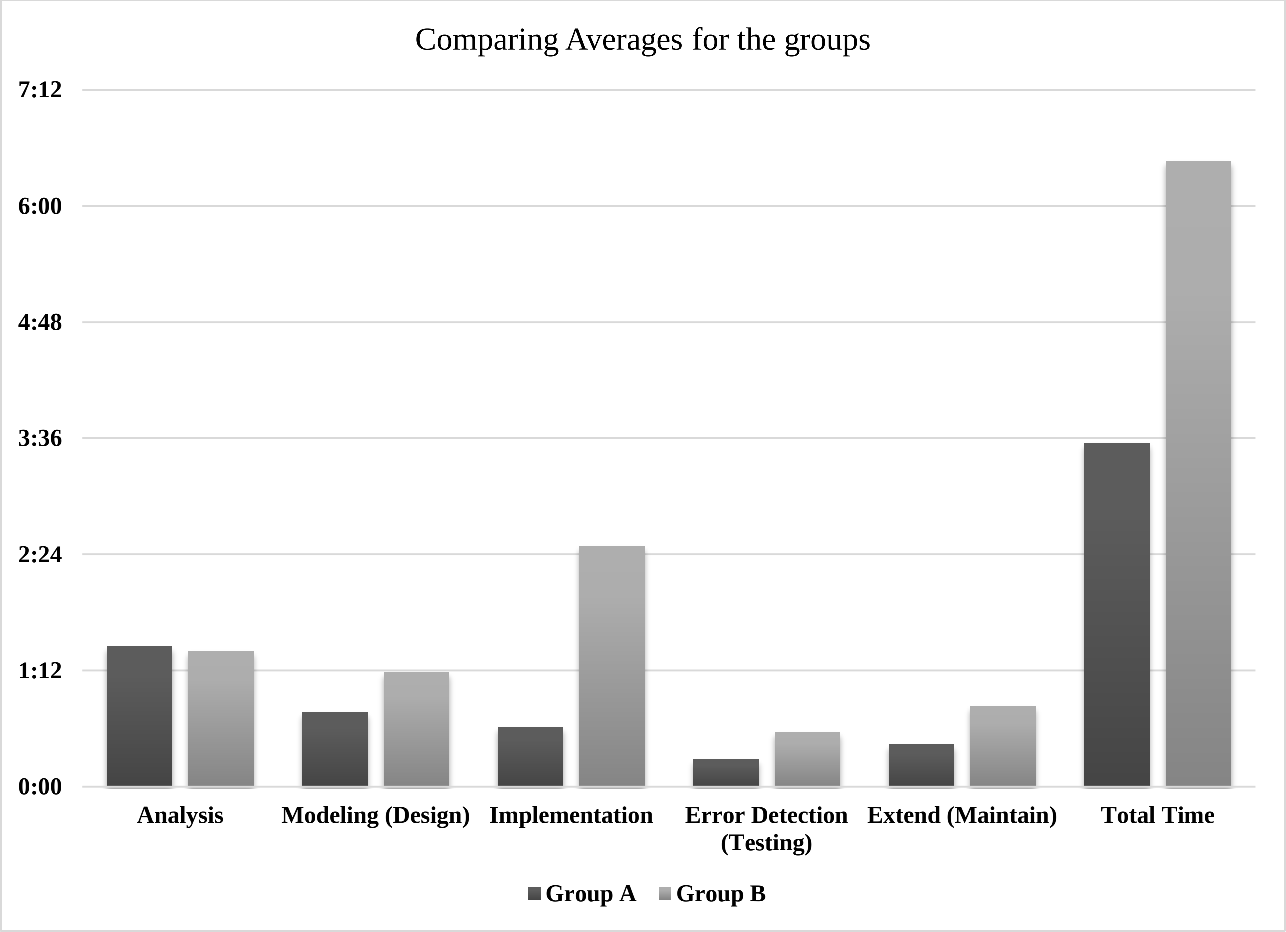
Initial Preparation Costs

Group	$C_{Teaching}$	$C_{ReadingDocs}$	C_{Demo}	$C_{TotalInit}$
A	2h:30mins	3h:30mins	1h:45mins	7h:45mins

The result of the average total cost and its elements in two groups and for different case studies

#	Case Study	Team	$C_{Analysis}$	$C_{Modeling}$	$C_{Implementation}$	C_{Error}	$C_{Maintainance}$	C_{Total}
1	Expert Finding	A	1h:36mins	0h:43 mins	0h:34 mins	0h:15 mins	0h:31 mins	3h:39 mins
		B	1h:40 mins	1h:11 mins	2h:53 mins	0h:34 mins	0h:54 mins	7h:12 mins
2	Stock Exchange	A	1h:47 mins	0h:53 mins	0h:44 mins	0h:21 mins	0h:24 mins	4h:09 mins
		B	1h:39 mins	1h:19 mins	2h:50 mins	0h:42 mins	0h:51 mins	7h:21 mins
3	Conference Management	A	1h:07 mins	0h:40 mins	0h:37 mins	0h:13 mins	0h:21 mins	2h:58 mins
		B	1h:13 mins	1h:03 mins	1h:52 mins	0h:31 mins	0h:47 mins	6h:26 mins
4	Hotel Reservation	A	1h:18 mins	0h:47 mins	0h:32 mins	0h:17 mins	0h:28 mins	3h:22 mins
		B	1h:06 mins	1h:12 mins	1h:40 mins	0h:29 mins	0h:49 mins	3h:16 mins
5	Average	A	1h:27 mins	0h:46 mins	0h:37 mins	0h:17 mins	0h:26 mins	3h:33 mins
		B	1h:24 mins	1h:11 mins	2h:29 mins	0h:34 mins	0h:50 mins	6h:28 mins

Evaluation – Result



- The results are prepared to be submitted to J. Soft Qual.

Conclusions & Future work

Conclusions

- For Semantic Web enabled MASs
- SEA_ML is provided using GMF, ATL, OCL, and Mofscript including:
  - Abstract syntax in 8 VPs (Ecore),
  - Graphical concrete syntax in GMF,
  - Code generation (M2M-ATL & M2T-Mofscript)
  - Semantics of the language with Helper and OCLrules
- SEA_L is also developed using
  - EBNF rules in Xtext and
  - Xtext Packages (e.g. Validation and Formatting Packages) for constraints
  - Xpand for code generation
- Evaluation

Conclusions – Future Work

The related study not covered here

•The Formal Semantics

- Model Checking using Alloy (IJCIS Journal, 2014)
- Attribute Grammar (Denotational Semantics), CSI Journal

Future work:

- Dynamic Semantics and Model@runtime for this domain
- Model Finding and Model Completion for the domain

Thank You!

&

Any Questions?