A Mapping from BPMN Model to JADEX Model

Sana Nouzri and AbdelAziz El Fazziki
Computer Science Department, University Cadi Ayyad, Morocco

Abstract: *The challenge for any enterprise is the evolution of its Information System (IS) to respond to unexpected requests. Face a complex IS, or any requested change; many enterprises live profound transformations. Today, agile IS must be tooled to provide the flexibility and adaptability that require the enterprise managed to remain competitive. This paper shows that Business Process Modeling (BPM) enriched by agent concepts is the best means for modeling and implementing ISs. The combination of these two technologies gives the idea for a new agent-oriented approach. This work focus on the development of Multi-Agent Systems (MAS) and a set of transformation rules. The transition from one model to another is ensured by a set of automated rules. The development process proposed is based on different meta-models (Business Process Modeling Language (BPMN), Agent Modeling Language (AML) and JADEX and automated transformation rules with Atlas Transformation Language (ATL). Finally, this work illustrates the proposal with a case study.*

Keywords: *IS, model driven architecture, MAS, BPM, automated transformation rules.*

Received December 10, 2012; accepted Septemper 9, 2013; published online April 17, 2014

1. Introduction

An enterprise is functionally agile when the organizational and strategic dimensions of Information System (IS) models collaborates to anticipate or to capture the change in order to dynamically compensate it and to integrate it. The efficient information systems depend on the coherence between strategy and organization. Strategic alignment perspectives and organizational alignment perspectives are the major keys of Business Process Modeling (BPM). The goal of strategic alignment of an IS is to help the enterprise to achieve its goals without being influenced by technical constraints, while organizational alignment aims to improve the operational functioning of the enterprise by the determination of activities to apply and actors who are responsible [16].

In general, when it is a representation of a process in which the decisions or availability of each actor play a key role in the execution of this process, the agent modeling is needed to enrich it. The business process agent modeling will be equipped with autonomy, interaction and cooperation. The emergence of agent technology has allowed IS to be viewed as a set of autonomous entities well structured and in interaction, rather than a set of objects and methods.

Multi-Agent Systems (MAS) paradigm offers an ideal framework the response to the requirements of complex IS. It constitutes an approach very rich and powerful in terms of modeling. The MAS modeling has faced many challenges. No agent-oriented methodology and formalism can cover alone all aspects of this field. Design and analysis phases are longer be covered while the implementation part is still theoretical. The main limitations related to agent-oriented methodologies have been resolved by Model Driven Architecture (MDA) [4]. Actually applying MDA requires the use of automation mechanisms for generating models as productive elements. Grace in automation, systems achieves a significant gain in productivity.

The MDA [9] allows IS developers to focus on the business and functional aspects of a system without considering the technical aspects only in the last phases of the development process. The construction of IS begins with BPM to satisfy all business needs. Business processes are usually expressed using the Business Process Modeling Language (BPMN) [10]. In order to, bring the concepts of autonomy, collaboration, interaction and cooperation, we define the business process concepts by the agent concepts Agent Modeling Language (AML) [14] and finally, we express the agent concepts compared to an implementation platform JADEX [13] in order to, enerate the code.

The objective of this work is to describe an agent oriented approach based on models. This approach, provides a complete development Lifecycle, based on a set of models and a set of automated transformation rules in order to automatically generate the code. The transition from one model to another is ensured by their meta-models. In this work, we will base on structural transformations. They are constituted by a set of rules, each expressing structural correspondence between source and target meta-model. The role of meta-models in model transformations is to define the possible structuring of the source and target models and provide the basis for the definition of transformation rules.

In this paper, we define a mapping from the domain model to BPMN model, from a BPMN model to AML agent model and from agent model to a JADEX specific platform model. The remainder of this paper is organized as follows: Section 2 presents the MAS modeling, section 3 describes an overview of the MDA architecture. The subject of the section 4 is the modeling approach description. Section 5 presents the overview of the meta-models. In section 6, we will illustrate the transformation rules. Section 7 presents the Atlas Transformation Language (ATL). Section 8 describes the process of BPMN to JADEX automated transformation. Section 9 illustrates the proposed approach with a case study. Finally, the section 10 concludes and presents the perspectives of this paper.

2. The Multi Agent System Modeling

The agent technology represents a new paradigm for the IS development. An agent is considered as an autonomous entity that has goals, evolves in an environment and interacts with it and with other agents. The object-oriented methods development are not directly applicable to MAS development. Hence, there is a need to extend or develop a new models, new methodologies and new tools adapted to the agent concept. The evolution of engineering techniques is in parallel with the evolution of development paradigms, after the revolution of object-oriented paradigm, we are therefore facing a new revolution that would be the agent-oriented paradigm.

Faced with the growing complexity of IS, the agent-oriented paradigm finds its utility in attacking the modeling of these systems. Indeed, the choice of a multi-agent approach is justified by the mechanisms of flexibility, adaptability, robustness, extensibility, modularity, speed and reliability. Generally, agents differ in how perceptions are related to actions. We traditionally distinguish two types of agents. The reactive agent that possesses neither mechanisms planning nor reasoning to act, but only mechanisms to react to events. Contrary to the reactive agent, the cognitive agent is an intelligent agent, with a “symbolic representation” of the environment from which he is able to make reasoning.

Agent within a system can be characterized by goals to achieve, plans and tasks to complete with other agents, which is the case of a MAS. A MAS is a system composed of autonomous agents designed to cooperate, interact and communicate to achieve a collective goal.

It is true that the MAS is the ideal solution to reduce the complexity of ISs operating in a dynamic and open environment, but the richness of MAS did meet many challenges. The big challenge is the MAS modeling in itself. No agent-oriented methodology and formalism can cover alone all aspects of this field. Construction of MAS requires the integration of different methods, techniques and architecture engineering for structuring the development process.

3. The MDA

MDA [9] is a software development approach, proposed and supported by the Object Management Group (OMG). The principle key of MDA is the use of models at different phases of the development of an IS. The implementation of MDA based on the development of appropriate meta-models, which are preliminary and need to write the transformation rules between different models.

The major objective of MDA is the development of persistent models independent of the technical details of the execution platforms (J2EE, Net, PHP or other). According to the MDA approach, the process of software development is piloted by models, which have qualities such as durability, productivity and consideration of execution platforms. More precisely MDA recommends the use of models for different phases of the development cycle of a system; we present the requirements models Computation Independent Model (CIM), analysis and design models Platform Independent Model (PIM) and implementation models Platform Specific Model (PSM). The MDA approach is not firm; it allows the use of other models such as the models of supervision, verification, optimization or test [18].

4. The Modeling Approach

Most agents oriented engineering techniques generally cover only the analysis and design phases; the implementation phase is tackled by the development platforms. In contrast to existing agent-oriented methodologies, our development approach ensures a complete development lifecycle, which is characterized by the separation of business logic and system implementation platforms. This approach is an agent-oriented approach based on a model for the development of IS; its strong point is the use of various means made available to developers to build MAS.

The implementation of this approach can be made only by the combination of the techniques that allow addressing different aspects of MAS development. The use of MDA allows covering all phases of development through the creation of a set of models and adopting transformations for the mapping between the models. In principle, this approach is based on BPM and agent modeling. The definition of business and agent concepts related to the system is flexible by using meta-modeling, a technique that is used to describe different aspects of MAS and their relationships.

4.1. The Approach Implementation

The implementation of the development approach is defined by the following steps:

- Definition of the development framework formed by meta-models and transformation rules.
- Automation of transformation rules using the defined meta-models.
- Using the framework developed by the creation of the first model and automatic generation of other models by applying transformation rules between these models.

4.2. The Development Process

Our approach focuses on the description of the MAS through three levels CIM, PIM and PSM. Each of these levels is based on a meta-model. The passage of each level to another is an automatic transformation process, which is based on a set of rules that model the experts knowledge of the application domain and of the designers in multi-agents. The development process allows the developer to create its development framework that is specific to its application. The development framework defines:

- Concepts and Business Properties.
- Concepts and Specific Properties of MAS.
- Transformation Rules between Business Concepts and Agents Concepts.
- Transformation Rules between the Agent Concepts and The Specific Platform Concepts (JADEX platform).

The Figure 1 illustrates the development process and the meta-models proposed.

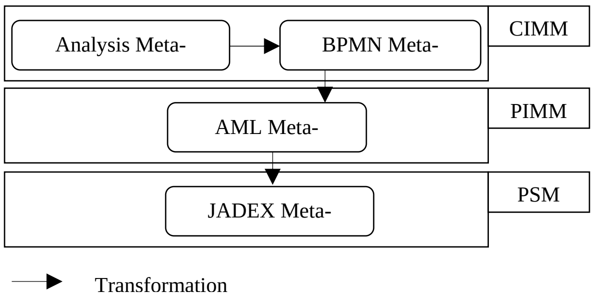


Figure 1. Development process.

4.3. The Domain Analysis

The analysis phase is the first phase of the development approach, it is necessary to define the requirement for BPMN modeling. The analyst needs first of all to identify enterprise domains and for each domain capture the functionality behind the system under development. In order to, do this, he needs to start thinking not in term of goal, but in term of what will the system need to do and who are the involved actors n each activity. Thus, the use case diagram helps to visualize the system, including its interactions with external entities. After defining the domain use cases, the work on the business process may start by structuring the domain use cases to define the corresponding business process [3].

In order to, properly finalize the realization of use cases and to separate the functionality for identifying the different actors types, we use three classes: Interface classes, control classes and entity classes of Boundary, Entity and Control (BEC) [8] class diagram. This is a first estimate of the classes that realize the application. Each use case must be made by the collaboration of these three types of classes. Figure 2 defines the interpretation of use cases by a BEC diagram as follows:

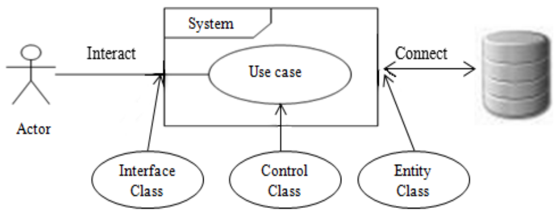


Figure 2. Relationship between use cases and analysis classes.

5. The Meta-Modeling

In our approach, we propose several meta-models. We start with the analysis meta-model described by use cases and BEC [8] class diagram, the business meta-model based on the BPMN [10] concepts to graphically represent business processes, the agent meta-model based on the AML [14] concepts and finally the implementation meta-model based on the JADEX [13] concepts. It is not necessary to rely on a unified framework for business processes meta-modeling and on a unified framework for agent paradigm, but it is sufficient to rely on the statements of the problem to design these two meta-model that seems best answered the sample application [3]. We use several transformations to achieve code generation and to ensure the transition from one model to another.

5.1. The BPMN Meta-Model

The BPMN language is a standard for business process modeling; it provides a graphical notation for specifying Business Process Diagrams (BPD) [15]. It is based on a flowcharting technique and is very similar to the UML activity diagram [17].

A process is divided into one or more pools, and it corresponds to one or more participants. A pool can be divided into several Lanes, representing the involved participants, organizational roles or departments. Each lane contains a part of the process in the form of atomic or composite activities associated with participants in the same control domain and are connected by sequence flows and message flows. Sequence flows describes the order in which activities must be completed while message flows describes the message exchange between pools [2].

5.2. The Agent Modeling Language Meta-Model

We give an introduction to the AML language and its different packages. The AML [14] is a semi-formal visual modeling language for specifying, modeling and documenting systems that incorporate concepts drawn from MAS theory.

We provide an overview of the package elements related to our work:

- *Type Agent*: Is an autonomous entity used to model an agent type that is able to act within their environment. The Type agent uses any kind of the UML class relationships.

- *Communication Message Payload*: Is a specialized UML class used to model the type of objects transmitted the message communications.
- *Belief*: Is a specialized mental class used to model a state of business objects, a proposal or other relevant information of the system.
- *Decidable Goal*: Is an abstract specialized class constrained mental class used to model goals i.e., conditions or states of business objects, which the main concern is their achievement or maintenance. Those goals can be used to represent objectives, needs, motivations, desires, etc.
- *Plan*: Is a specialized class constrained mental class which corresponds to a UML activity, used to model capabilities of mental semi-entity types.

6. The Transformation Rules

Modeling a business process by an SMA is representing all the business process elements by the agent paradigm. The activities will be modeled in terms of goals, plans and will represent the agent behaviors. Communications protocols play an important role if the realization of an activity requiring the collaboration of several agent [2].

The proposed BPAM suggests a classification of agents into two categories: Cognitive agents and reactive agents. The approach is performed in two steps. The first one uses a mapping from the BEC diagram into agent types to define MAS. The second one uses a mapping from BPMN model to AML agent concepts. The result of this mapping is the agent behavior models. The Figure 3 presents an overview of the different mapping approach.

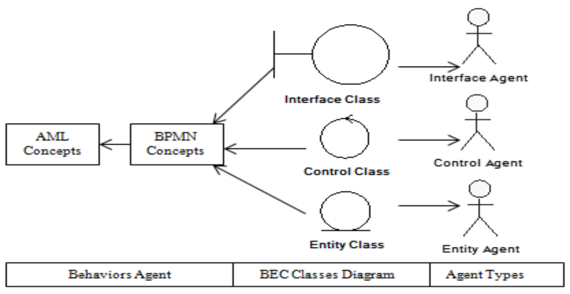


Figure 3. Different level of mapping.

6.1. A Mapping from BPMN Model to AML Model

In the BPMN language, a process is a set of pools and pool presents a sequence of activities related to an agent or a group of agents. In the BPMN definition, we distinguish two types of pools: A simple pool and structured pool which is a set of lanes. For each simple pool or a lane we derive an agent's behavior, this behavior is defined by the specification of its plans and actions. Agents have a set of plans that are selected for execution. These plans are based on their goals and their states. In this paper, we propose to deduce goals and plans of an agent from the BPMN diagram by using a set of transformation rules. We consider a plan as an activity. The main transformation rules are presented in the Table 1.

Table 1. Mapping from BPMN to AML.

BPMN Concepts	AML Concepts
Process	SMA
Pool	Group of Agent Type
Lane	Agent Type
Sub Process	Plan
Task	Action (Extention)
Intermediate Event	Communication Message Payload
End Event	Decidable Goal
Data Object	Belief

6.2. A Mapping from AML Model to JADEX Model

In this section we define the transformation rules between the AML package elements and JADEX elements defined in the ADF file. Table 2 defines the correspondences between AML and JADEX concepts.

Table 2. Mapping from BPMN to AML

AML Concepts	JADEX Concepts
Agent Type	Name of the ADF File
Belief of Agent Type	Belief
Decidable Goal of Agent Type	Achievegoal
Plan of Agent Type	Plan
Communication Message Payload	Action (stereotype) of Agent Type
Start Event Message	Message Event
First Plan of Agent Type	Decidable Goal
Action of Agent Type	Function Implemented in the Plan Java Class

7. The Atlas Transformation Language

There exist several approaches for the transformation rule specification. The programming approach based on object-oriented programming languages with interfaces for manipulating models. The template approach based on specific language to define a target model parameterized by a source model. The modeling approach based on the MDA principles to model the transformation rules. Often, all these approaches realize any transformation; just the way of the rule making that differs. The programming approach and the template approach are the easiest compared to the modeling approach, having regard to the celebrity languages that use them. These languages require no learning contrary to the modeling approach, is more complex, but the most promising because it offers solutions to sustainability problems of the transformations and their reuse [18].

In this work, we chose the modeling approach because its principle is to represent model transformations in an appropriate way. Thus, these models are sustainable, productive and are defined adequately to a specific platform. The idea is to apply the concepts of model driven engineering to the model transformations themselves. The realization of this approach is based on the standard Meta Object Facility (MOF) 2.0 Query/View/Transformation (QVT) [12] which aims to define the meta-model allowing the development and structuring models of models transformation. The ATL is one of the responses to the QVT and is presently one of the available existing QVT languages with a quite broad and expanding user community.

ATL [6] is the Atlas INRIA and LINA research group's response to the OMG's MOF 2.0 QVT RFP [11]. It is a model transformation language defined to perform general transformations within the MDA framework and is specified both as a meta-model and as a textual concrete syntax. ATL is a hybrid of declarative and imperative programming. Transformations can be defined between arbitrary MOF 1.4 [11] or Eclipse Modeling Framework (EMF) [1] based meta-models by specifying rules that define how source model elements are matched and navigated to create and initialize the elements of the target model.

7.1. The ATL Model Transformation Process

Transitions between Models are accomplished conformly to required meta-models by using the ATL transformations that are conforming to the MDA principles.

The ATL transformation engine currently provides a support for two existing technologies: the MOF 1.4 [11] defined by the OMG and the Ecore meta-meta-model defined by the EMF [1]. Figure 4 illustrates the full model transformation process in the context of the MDA framework.

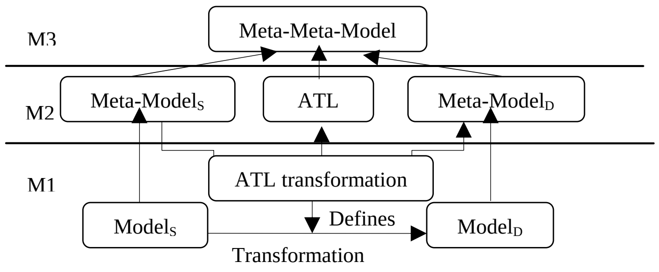


Figure 4. ATL model transformation.

8. The BPMN to JADEX Automated Transformation

Generally, the BPM defines a system global view. It can be used as a basis for the development of agents behavior. Thus, by identifying for each agent: Its plans, goals, interactions, statements and resources.

Normally, the derivation of the agents behavior from a BPMN diagram is highly dependent on the type of agents and how their behaviors are implemented in a specific platform. This derivation is essentially deduced from a set of rules based on the concepts of the agent meta-model and BPMN meta-model.

To transform a BPMN diagram into an AML model, and an AML model into JADEX model, we adopt the ATL [6]. To become an instance of the MOF or ECORE based BPMN meta-model, the source model has first to be injected into the BPMN meta-model. The mapping includes several phases.

8.1. The XML Generator Phase

First of all, it is necessary to recover an exploitable file of the BPMN graphical model. In fact, we try to use an exploitable definition of a BPMN process to be able to translate it. This requires that the BPMN editor can generate a file that defines the process and that is described using a non-proprietary format, but standardized and exploitable such as XML. We chose the intalio designer [5], the tool allows to design processes according to BPMN grammar and to generate an XML definition from processes designed graphically. Figure 5 illustrates the transformation process from the graphical model to XML model.

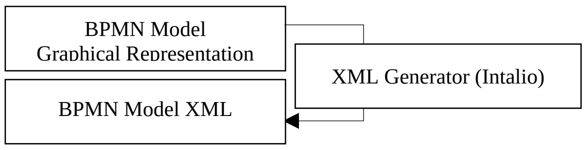


Figure 5. XML generator.

8.2. The XML Injector Phase

The XML Injector is included in the ATL Development Tools (ADT), indeed ATL cannot directly manipulate an XML file. But, it can use an XML model which is conform to the XML meta-model. Therefore, the injector is used to directly transform an XML file (a file with a. XML) into an XML model (a file with the extension. ecore or .XMI). We try to produce a BPMN model which is conform to the XMI/ ECORE based XML meta-model from an XML representation of a BPMN model. Figure 6 illustrates the transformation process from XML file to XML model.

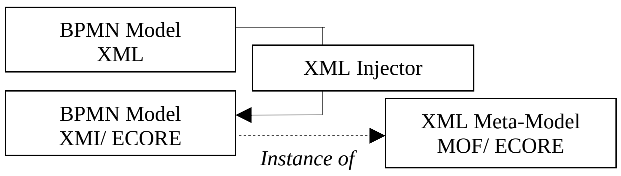


Figure 6. XML injector.

8.3. The ATL Transformation XML2BPMN Phase

We transform the resulting BPMN model conforming to the XML meta-model into a BPMN model conforming to the BPMN meta-model using an ATL transformation XML2BPMN file. The execution of this file requires a set of pre-requisites.

- The resulting BPMN model obtained by using the XML injector.
- The XML standard meta-model: This is the XML. ECORE file which describes the structure of an XML document.

- The BPMN meta-model by using the EMF editor.

Figure 7 illustrates the transformation process from XML model to BPMN model.

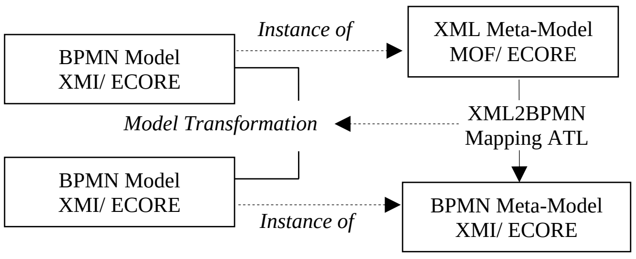


Figure 7. XML2BPMN mapping.

8.4. The ATL Transformation BPMN2AML Phase

This phase concerns the transformation of the resulting BPMN model to the target AML model using the ATL transformation file (BPMN2AML). The execution of this file requires a set of pre-requisites:

- The Adequate Resulting BPMN Model.
- The BPMN Meta-Model: It is the same meta-model identified in the previous transformation.
- The AML Meta-Model: We opt for the realization of this meta-model by using the EMF editor. This meta-model is written in ECORE and described in the UML language in the version 1.3. The use of a UML profile associated with the transformation rules allows having stereotyped classes that better reflect the specificity of the target model AML model. For this reason, we have included the AML profile with the UML meta-model.

Figure 8 illustrates the transformation process from BPMN model to AML model.

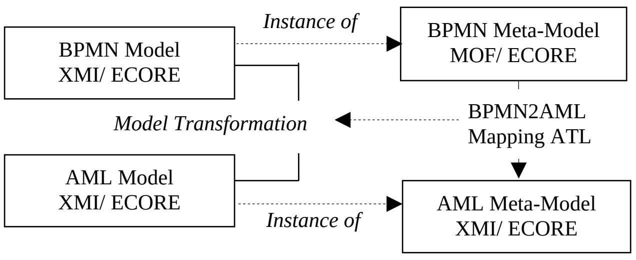


Figure 8. BPMN2AML mapping.

8.5. The ATL Transformation AML2JADEX Phase

The final step is the transformation of the resulting AML model to JADEX model using the ATL transformation files AML2JADEX and AML2JAVA. The execution of these files requires a set of pre-requisites.

- The Resulting AML Model: This is the result file of the previous step.
- The AML Meta-Model: It is the same meta-model identified in the previous transformation.
- The JADEX Meta-Model: We opt for the use of EMF editor. In reality, there is not a single meta-model, but in fact two meta-model, the ADF meta-model and the Java meta-model. The two meta-models are written in ECORE and describe respectively the ADF file and the Java programming language.

Figure 9 illustrates the transformation process from AML model to JADEX model.

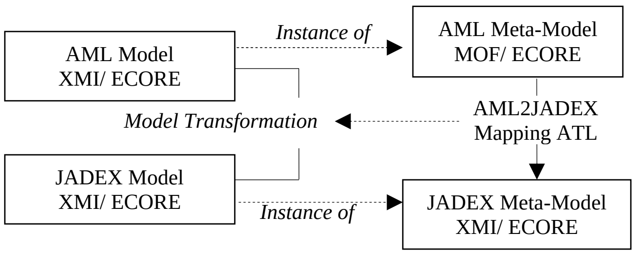


Figure 9. AML2JADEX mapping.

8.6. The Code Generation

After the execution of the previous ATL transformations AML2JADEX phase we get two files:

- The ADF File (JADEXModel.xml): Contains the structure and the agent concepts of the system.
- The Java Model (JAVAModel.xml): We use a STARUML UML editor to import Java model (JAVAModel.xmi) and to generate the java code.

9. A Case Study

The case study to validate the proposed approach is a conference Paper Review and Submission System (PRSS) [7]. This kind of system involves a number of collaborative processes. These processes may be separate and independent and are linked to a session or have a short life cycle. The Review and Submission System is an appropriate system to be described by a business process language which is simple and comprehensible by everyone. The PSR system's objective is to reduce the conference steering committee tasks and to allow a high collaboration among the conference organizing actors and participants and to provide basic services to authors, reviewers and track chairs.

9.1. The Domain Analysis

The conference organization is the main objective of the PRSS. Organizing a conference requires four phases:

- *The First Phase*: Concerns the steering committee information and identification. The steering committee members can connect and edit their profile by using their own password.

- *The Second Phase*: Concerns the bid, where reviewers are asked to specify the degree their expertise for each paper.
- *The Third Phase*: Concerns the paper assignment, where a track chair have in charge to assign papers to reviewers. The chair must be based on the reviewers bid while taking into consideration the number of reviewers by a submission and the number of submissions by a reviewer.
- *The Fourth Phase*: Is devoted to papers review and submission. The submission is done in two steps, the abstract submission and the full submission.

During these four phases of the overall process, the conference information like submission deadlines, review, bidding and tracks managing; this information is the most relevant for the conference management. The objective of this case study is to improve the business process agent modeling. For this reason, we chose the paper assignment phase, which is taken in charge by the conference chair. Figures 10 and 11 present the use case diagram and BEC digaram which describe the functionality of the conference chair.

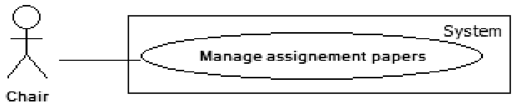


Figure 10. Use case diagram of chair actor.

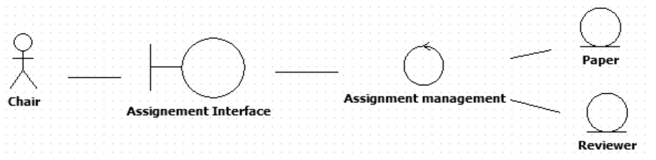


Figure 11. The BEC diagram of the conference chair.

9.2. The BPMN Model

The overall process of the PRSS is the conference management. This process consists of a set of sub processes. The PRSS is characterized by messages exchange among several actors. These messages may invoke a new instance of a sub process. The “assignment” sub process and the message exchange between the different participants are described in the BPMN language. Figure 12 shows a part of the “assignment” sub process. According to the BEC diagram, the activities of each sub-process requires the collaboration of three types of classes: Interface, control and entity:

- *Interface Manager*: Who is firstly responsible for recovering requests and external data and secondly to direct requests to appropriate actors.
- *Manager Chair*: Who is responsible for managing the activities of the sub-process “assignment”.
- *Database Manager*: Which is responsible for managing the database.

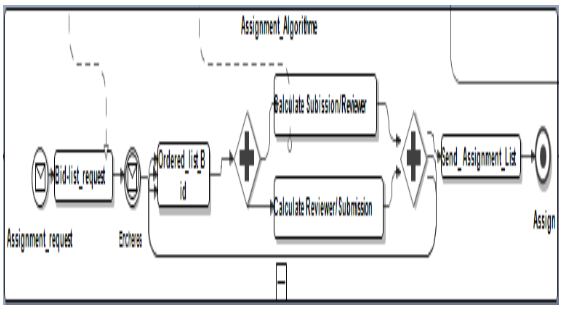


Figure 12. The BPMN diagram, assignment subprocess.

9.3. The AML Model

The interface and entity classes are represented by reactive agents; contrary to the “chair manager” class which is represented by a cognitive agent.

To perform the assignment sub process, the “chair manager” agent uses the bid list. The objective is to build a submission list for each reviewer where the list must respect the number of reviewers per submission and the number of submissions per reviewer. The following figure presents the “assignment” sub-process.

Figure 13 shows an overview of the AML agent model file automatically generated after applying the ATL transformation BPMN2AML. It is edited with the STARUML editor as agent diagram. However, STARUML does not automatically organize the elements of an agent diagram. A manual effort is required for this task.

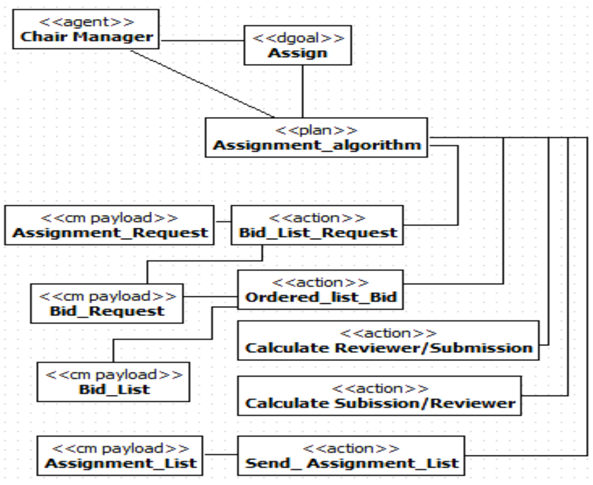
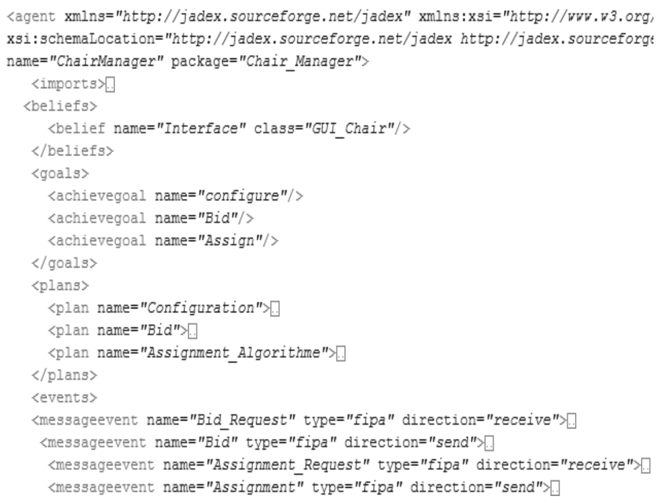

<<agent>> Chair Manager
<<dgoal>> Assign
<<plan>> Assignment_algorithm
<<cm payload>> Assignment_Request
<<action>> Bid_List_Request
<<cm payload>> Bid_Request
<<action>> Ordered_list_Bid
<<action>> Calculate Reviewer/Submission
<<cm payload>> Bid_List
<<action>> Calculate Subission/Reviewer
<<cm payload>> Assignment_List
<<action>> Send_ Assignment_List

Figure 13. The AML model of “chair manager” agent.

9.4. The JADEX Model

Figure 14 below shows an overview of the results generated after applying the ATL transformation of the file AML2JADEX, generating an XML file which describes the concepts of the “chair manager” agent according to the JADEX logic. This file is used by the UML Eclipse [1] editor.

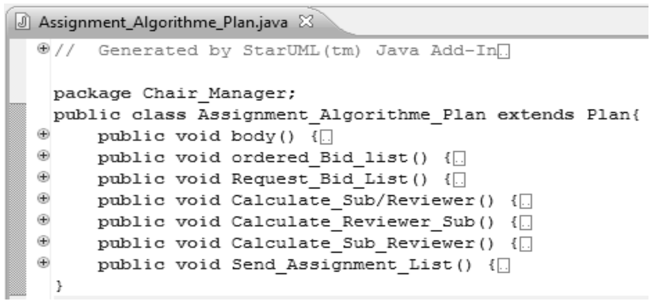


```
<agent xmlns="http://jadex.sourceforge.net/jadex" xmlns:xsi="http://www.w3.org/
xsi:schemaLocation="http://jadex.sourceforge.net/jadex http://jadex.sourceforg
name="ChairManager" package="Chair_Manager">
   <imports>
  <beliefs>
     <belief name="Interface" class="GUI_Chair"/>
   </beliefs>
   <goals>
     <achievegoal name="configure"/>
     <achievegoal name="Bid"/>
     <achievegoal name="Assign"/>
   </goals>
   <plans>
     <plan name="Configuration">
     <plan name="Bid">
     <plan name="Assignment_Algorithme">
   </plans>
   <events>
   <messageevent name="Bid_Request" type="fipa" direction="receive">
    <messageevent name="Bid" type="fipa" direction="send">
     <messageevent name="Assignment_Request" type="fipa" direction="receive">
     <messageevent name="Assignment" type="fipa" direction="send">
```

Figure 14. An ADF file of "chair manager" agent.

9.5. The Generated Code

The result obtained after applying the ATL transformation AML2JAVA gives a xml file that contains the implemented plans corresponding to JADEX model. We import this file by the STARTUML editor and we automatically generate the corresponding code as described in Figure 15. A manual intervention is required to complete and execute the code.



Assignment_Algorithme_Plan.java

```
//  Generated by StarUML(tm) Java Add-In

package Chair_Manager;
public class Assignment_Algorithme_Plan extends Plan{
    public void body() {
    public void ordered_Bid_list() {
    public void Request_Bid_List() {
    public void Calculate_Sub/Reviewer() {
    public void Calculate_Reviewer_Sub() {
    public void Calculate_Sub_Reviewer() {
    public void Send_Assignment_List() {
}
```

Figure 15. The generated code from JADEX model.

10. Conclusions

In this paper, we presented a mapping from BPMN model to AML model and a mapping from the AML model to JADEX model; we presented an original approach that associates the agent paradigm, the MDA and BPMN. The proposed methodology can make the development of complex systems better aligned with changing business needs easier and less costly. In this paper, we presented a definition of agent oriented approach which can be considered as a set of components which produces a response to the development challenges that must solve the requirements of ISs development. Our main objective is the implementation of a complete agent oriented approach and the automation of transformation rules. To this end, it seems to be necessary after defining a set of the principal transformation rules for the mapping between BPMN, AML language and JADEX platform, to solve enough problems, to get an adequate result.

First of all, it is necessary to explore for the workflow and data object. Therefore, we will also identify any further requirements that are needed for the agent technology in order to capture the desired functionality and behaviors.

References

[1] Budinsky F., Steinberg D., Ellersick R., Merkes E., Brodsky A., and Grose J., *Eclipse Modeling Framework*, Addison Wesley, Boston, USA, 2003.

[2] El Fazziki A., Nouzri S., and Sadgal M., " An Agent-Oriented Information System: A Model Driven Approach," *International Journal of Computer Applications*, vol. 47, no. 4, pp. 1-8, 2012.

[3] El Fazziki A., Nouzri S., and Sadgal M., " An Agent Oriented Information System: An MDA Based Development," *International Journal of Computer Science Issues*, vol. 9, no. 1, pp. 40-48, 2012.

[4] Elammari M. and Issa Z., "Using Model Driven Architecture to Develop Multi-Agent Systems," *the International Arab Journal of Information Technology*, vol. 10, no. 4, pp. 349-355, 2013.

[5] Guo H., *Introduction of Intalio|Designer 6.0.1 and Intalio|Server 6.0.1*, available at: http://www.idi.ntnu.no/emner/tdt4250/Slides/IntalioIntroduction2009.pdf, last visited 2009.

[6] Jouault F., Allilaire F., Bézivin J., Kurtev I., "ATL: A Model Transformation Tool," *Science Computer Programming*, vol. 72, no. 1, pp. 31-39, 2008.

[7] Kirchberg M., S¨orensen O., and Thalheim B., "A BPMN Case Study: Paper Review and Submission System," available at: http://subs.emis.de/LNI/Proceedings/Proceedings154/gi-proc-154-375.pdf, last visited 2009.

[8] Lo P., "Designing boundary classes," available at: http:// www.peter-lo.com/ Teaching/M8748/ L17.pdf, last visited 2007.

[9] Miller J. and Mukerji J., "MDA Guide Version 1.0.1, OMG," available at: http:// staffwww.dcs.shef.ac.uk/people/A.Simons/remodel/papers/MDAGuide101Jun03.pdf, last visited 2003.

[10] Muehlen M. and Indulska M., "Modeling Languages Processes and Business Rules: A Representational Analysis," *Journal of Information Systems*, vol. 35, no. 4, pp. 379-390. 2010.

[11] Object Management Group, Meta Object Facility (MOF) 1.4, Final Adopted Specification formal/02-04-03, available at: http://www.omg.org/ cgi-bin/ doc?formal/ 02-04-03.pdf, last visited 2002.

[12] Object Management Group, *Query/Views/ Transformation* (QVT) RFP, Request for Proposal ad/02-04-10, 2002.

[13] Piunti M. and Studiorum A., “Programming BDI Agents in Jadex,” Università di Bologna-DEIS. 2008.

[14] Radovan C. and Ivan T., “Agent Modeling Language: Language Specification. Version 0.9,” *Technical report*, Whitestein Technologies, 2004. http://www.whitestein.com/pages/solutions/meth.html.

[15] Richard C., “An XML Representation for Crew Procedures,” available at: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20050202022_2005202152.pdf, last visited 2004.

[16] Thierry L., “L'agilité du système d'information, ça se prépare,” available at: http://www.cio-online.com/actualites/lire-l-agilite-du-systeme-d-information-ca-se-prepare-3289.html, last visited 2010.

[17] White S., “Process Modeling Notations and Workflow Patterns,” available at: http://www.omg.org/bpmn/Documents/Notations_and_Workflow_Patterns.pdf, 2004.

[18] Xavier B., *MDA En Action Ingénierie Logicielle Guide Par les Modèles*, éditions eyrolles, 2005.

Sana Nouzri is a member in the research laboratory (computer system engineering) of the Computer Science Department at the Faculty of Semlalia (Marrakech, Morocco) for the preparation of her thesis; she worked on agent-oriented approach based on models for the development of information systems. Her current research interests include information system engineering, agent based system, agent-oriented methodologies, model driven architecture and business process modeling.

Aziz El Fazziki is a Professor of computer science at Marrakech University, where he has been since 1985. He received an MS from the University of Nancy (France) in 1985. He received his PhD in computer science from the University of Marrakech in 2002. His research interests are in software engineering, focusing on information system development. In the MDA arena, he has worked on agent based systems, and service oriented systems an decisional systems. He is co-author agent based image processing, and is the author of over fifty papers on software engineering.