MDD-based agent-oriented software engineering for ubiquitous deployment

Jorge Agüero, Miguel Rebollo, Carlos Carrascosa,Vicente Julián
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera S/N 46022 Valencia (Spain)
Email: {jaguero, mrebollo, carrasco, vinglada}@dsic.upv.es

***Abstract*—This work presents a Model Driven Development (MDD) approach to agent-oriented software engineering in order to design and deploy application prototypes in a fast and simple way. This approach is specifically addressed to systems including agents that must be executed on mobile or embedded devices. The user will design the system for different platforms by means of unified agent models (UML-like). There will exist different automatic transformations to obtain the specific code for different target platforms from these unified models. On this way, the deployment process of mobile/embedded agent-based applications is simplified.**

***Index Terms*—Multi-Agent Systems, Agent-Oriented Software Engineering.**

I. Introduction

MDD is a technique allowing to obtain code for different execution platforms from models by means of consecutive transformations. Currently, this approach is being adopted in several software developing areas such as Multi-Agent Systems (MAS), to improve not only the development process, but also the quality of the software based on agents. This work presents a method to obtain embedded agents code to be executed in different mobile platforms. That is, to design ubiquitous applications with agents using models or abstract concepts forgetting the implementation details and any platform detail (using a set of visual components generated for the tools supporting the MDD). After that, the specific embedded agent for the platform where it must be executed is generated by transformations. In this way, a non-expert programmer will be able to develop systems with ubiquitous agents, reducing the gap between design and implementation. This document is structured as follows. Section II shows how to apply the MDD approach to develop ubiquitous agents according to the approach of the present work. Finally, some conclusions are presented in section III.

II. MDD for MAS

The purpose of MDD is to create models legible by computers that can be understood by automatic tools to generate templates, proof models and even code, integrating them in multiple platforms [4].

From the viewpoint of the design of agent oriented systems, applications development consists in the obtaining of the agent code that could be executed in many platforms. That is, to concentrate the development of the application from an unified agent model and to apply different transformations to get implementations for different platforms. Currently, most common methodologies for multi-agent systems have identified a set of models that specify their characteristics. This abstraction level is known as Computation Independent Model (CIM). These models can be adjusted as MDD models that specifies the concepts of the MAS, as roles, behaviors, tasks, interactions or protocols. The models can be used to model a MAS without focus on platform-specific details and requirements, as a Platform Independent Model (PIM), which represents the system functionalities without consider the final implementation platform. After that, it is possible to transform PIM models into Platform Specific Models (PSM). Figure 1 shows possible relationships between the concepts of different MDD models and their transformations [2].

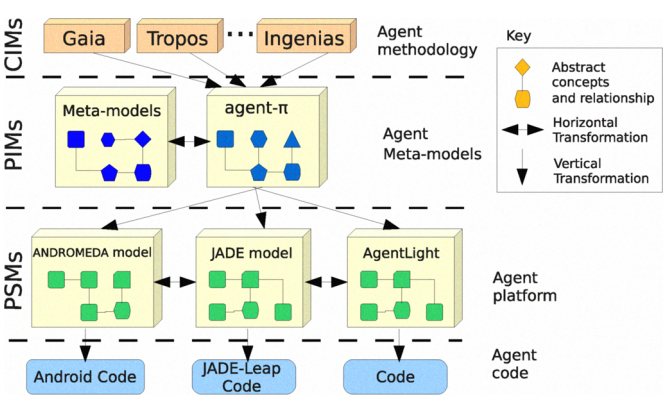

Fig. 1. MDD for MAS

A. Using MDD to develop embedded agents

The proposed design of embedded agents for mobile or, more generally, ubiquitous environments follows the approach offered by the MDD technology. The design process starts trying to modelize the agents using the abstract concepts and relationships established in the agent meta-model. The unified agent meta-models proposed in this work is called *agent-π*, *agent-Platform Independent* (detailed in Agüero et al [1]).

Next, the process continues transforming agent meta-models (PIM) into an agent model that depends of the chosen target

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.6980
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.6980

platform (PSM), by defining a set of mapping rules. After this, it is necessary to convert the model into code templates. This code can be optionally combined with hand-made code written by the user. See Figure 2 to illustrate the process (dotted lines represent the correlation between the concepts of the two meta-models).

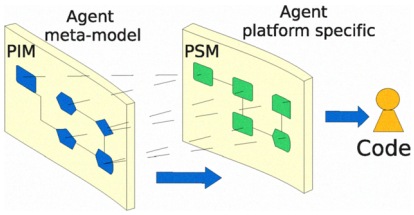

Fig. 2. Implementing an agent using MDD transformations

B. Development process

According to the process previously explained, the agent design is formed by a set of steps or phases (mainly transformations) that will obtain the executable code. In order to do these steps, a set of of tools, which support the process, are required. The tools employed at each stage of the design can be summarized as follows (see Figure 3):

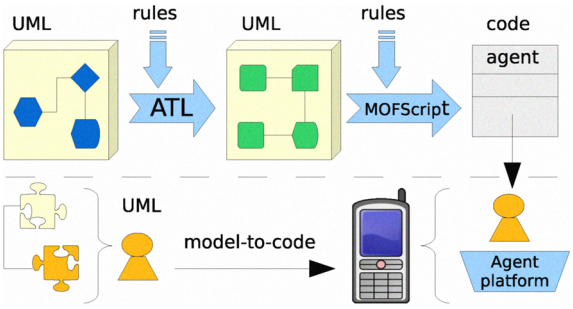

Fig. 3. Model-to-code transformation

- Step 1: to create diagrams (through graphical tools) which model the different behaviors, tasks, interactions, etc. of the agents. To perform this step, the Eclipse IDE with a set of *plugins* is employed. These plug-ins are mainly *EMF*, *Ecore*, *GMF* and *GEF*, which allow the user to draw the models that represent the agent.
- Step 2: to select which platform the user wants to execute the agent. This phase corresponds with the PSM model definition of each agent. To do this, it is necessary to apply a model-to-model transformation (PIM-to-PSM). This is done using the Eclipse IDE and the ATL *plug-in* incorporating the appropriated set of transformation rules.
- Step 3: to apply a transformation to convert the model into the agent code. To do this, we must use a PSM-to-code transformation. In this case, we use *MOFScript* which is an Eclipse plug-in that uses templates to do the translation.

Summarizing, Figure 4 illustrates how it is possible to obtain code for different agent platforms from unified models (*agent-π*). In this first version, the work has been focused to generate code for two agent platforms (study of agent model transformations into two mobile agent platforms): ANDROMEDA[1] [2] and JADE-Leap[2] [3]. JADE-Leap, which is one of the most known agent platforms for mobile devices and the ANDROMEDA platform, specifically designed to execute agents over the Google's Android OS.

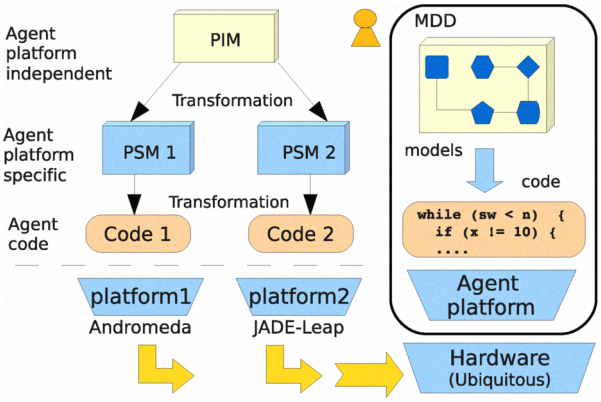

Fig. 4. Transformation process to different platforms

III. CONCLUSIONS

This paper presents the application of the ideas proposed by the MDD for the design of embedded agents into mobile device platforms. So, this work has checked the simplification of the agent design, since that some implementation details are hidden to the designer. The automatic transformations allow generating code templates for different platforms using a unified agent model (favoring inter-operability). Future work will focus on developing new automatic transformations to other embedded agent platforms, simplifying the creation of real heterogeneous ubiquitous systems.

IV. ACKNOWLEDGMENT

This work was partially supported by the Spanish government under grant CSD2007-00022 and under FEDER grant TIN2006-14630-C0301 project and in part by the Valencian Government under grant PROMETEO 2008/051

REFERENCES

[1] J. Agüero, M. Rebollo, C. Carrascosa, and V. Julián. Towards on embedded agent model for android mobiles. In *The Fifth Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (Mobiquitous 2008)*, volume CD Press, ISBN: 978-963-9799-21-9, pages 1–4, Dublin, Ireland, 2008.

[2] J. Agüero, M. Rebollo, C. Carrascosa, and V. Julián. Agent design using model driven development. In *7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS2009)*, volume 55, ISBN 978-3-642-00486-5, pages 60–69, Salamanca, Spain, 2009.

[3] F. Bergenti and A. Poggi. Leap: A fipa platform for handheld and mobile devices. In *Intelligent Agents VIII, Proceedings of the Eighth International Workshop on Agent Theories, Architectures, and Languages (ATAL-2001)*, 2001.

[4] A. Kleppe, J. B. Warmer, and W. Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Professional, 2003.

[1] http://users.dsic.upv.es/grupos/ia/sma/tools/Andromeda
[2] http://jade.tilab.com

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2009.6980
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2009.6980