A Domain Engineering Process for Developing Multi-agent Systems Product Lines

(Extended Abstract)

Ingrid Nunes
PUC-Rio, LES
Rio de Janeiro, Brazil
ioliveira@inf.puc-rio.br

Uirá Kulesza
Federal University of Rio Grande do Norte (UFRN)
Natal, Brazil
uir@dimap.ufrn.br

Camila Nunes,
Carlos J.P. Lucena
PUC-Rio, LES
Rio de Janeiro, Brazil
{cnunes,lucena}@inf.puc-rio.br

ABSTRACT

Multi-agent Systems Product Lines (MAS-PLs) have emerged to integrate two promising trends of software engineering: agent-oriented software engineering and software product lines. In this paper, we propose a domain engineering process to develop MAS-PLs, built on top of agent-oriented and software product line approaches.

Categories and Subject Descriptors
I.2.11 [Distributed Artificial Intelligence]: Multiagent systems; D.2.8 [Software Engineering]: Reusable Software—Domain engineering

General Terms
Design, Documentation

Keywords

1. INTRODUCTION

Agent-oriented software engineering has emerged as a new software engineering paradigm to help on the development of complex and distributed systems, based on agent abstraction. Several works were proposed in this direction, such as methodologies and modeling languages. However, most of agent-oriented methodologies do not take into account the adoption of extensive reuse practices, which have been widely used in the software engineering context to provide reduced time-to-market and lower development costs.

In the context of software reuse, the concepts of system families and software product lines (SPLs) have gained a significant popularity throughout the software industry and research community, leading to the emergence of a new field called software product line engineering (SPLE). Although many SPL methodologies have been proposed, they do not detail or barely detail the modeling and documentation of SPLs that take advantage of agent technology. Only some recent research [3, 9] has explored the integration of SPL and Multi-agent System (MAS) technologies, by incorporating their respective benefits and helping the industrial exploitation of agent technology.

This paper presents a domain engineering process for developing MAS-PLs. Our approach is the result of an investigation of how current SPL, MAS and MAS-PL approaches can model MAS-PLs. Based on this experience, we propose our process, built on top of some MAS and SPL approaches.

2. THE DOMAIN ENGINEERING PROCESS

SPLE is in general organized in two main processes: domain engineering and application engineering. Domain engineering is the process of SPLE in which the commonalities and variabilities of the SPL are identified, defined and realized. Our proposal is a domain engineering process that defines the stages and their respective activities to develop MAS-PLs. It aggregates some activities that are specific to model software agents and their variabilities. Table 1 summarizes the stages and activities that compose our process, and the artifacts produced in each one of them as well.

Our domain engineering process was conceived by the integration of existing works in the context of MASs and SPLs, which are: (i) PLUS [4] method; (ii) PASSI [1] methodology; and (iii) MAS-ML [2] modeling language. In addition, we propose additional adaptations and extensions for them.

We have investigated the use of current SPLs and MAS-PL approaches for modeling and documenting agency features in MAS-PLs [7]. SPL methodologies provide useful notations to model the agency features. However, none of them completely covers their specification. Particularly, the PLUS (Product Line UML-based Software engineering) [4] approach was very useful for documenting agency features. PLUS provides a set of concepts and techniques to extend UML-based design methods and processes for single systems to handle SPLs. As a consequence, this approach was the base for the elaboration of our process.

Nevertheless, agent technology provides particular characteristics that need to be considered in order to take advantage of this paradigm. In order to model agency features in MAS-PL, we have adopted the phases of the System Requirements model of PASSI [1] methodology, which is agent-oriented. We made some adaptations in these phases
and they were incorporated into the domain analysis stage. The incorporated phases with our extensions are described in Table 2. PASSI follows the guideline of using standards whenever possible; and this justifies the use of UML as modeling language. This helped us to incorporate some PLUS notations into PASSI diagrams.

### Table 2: Our extensions to the PASSI Approach.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent Identification</td>
<td>Only use cases in &lt;&lt;agency feature&gt;&gt; stereotypes packages are distributed among agents</td>
</tr>
<tr>
<td>Role Identification</td>
<td>Use of UML 2.0 frames (crosscutting features)</td>
</tr>
<tr>
<td>Task Specification</td>
<td>Use of UML 2.0 frames (crosscutting features)</td>
</tr>
</tbody>
</table>

### 3. CONCLUSION AND FUTURE WORK

In this paper, we presented a domain engineering process to develop MAS-PLs. Our approach incorporates activities and notations of different well-succeeded works in the context of SPLs and MASs: PLUS provides notations for documenting variability; PASSI methodology diagrams are used to specify agency features; and MAS-ML is the modeling language used in the domain design stage. We also proposed some adaptations and extensions to these approaches to address agency features. We have developed our process with the experience of two cases studies: the OLIS [5], a product line of web applications that provide personal services to users, and the ExpertCommittee [6], a conference management systems product line.

We are currently working on the development of other case studies to evaluate our process. In addition, we aim at formalizing our process using a process modeling language.

### 4. REFERENCES


