Extending PASSI to Model Multi-agent Systems Product Lines

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
uira@dimap.ufrn.br

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

ABSTRACT

Multi-agent System Product Lines (MAS-PLs) have emerged to integrate software product lines (SPLs) and agent-oriented software engineering techniques by incorporating their respective benefits and helping the industrial exploitation of agent technology. In this paper, we present a new approach for modeling MAS-PLs, focusing the domain analysis stage. Our approach is based on PASSI methodology and incorporates some extensions to address agency variability.

Categories and Subject Descriptors

D.2.13 [**Software Engineering**]: Reusable Software; I.2.11.d [**Artificial Intelligence**]: Multiagent systems

General Terms

Multi-agent Systems Product Lines

Keywords

Multi-agent Systems, Software Product Lines, Methodology

1. INTRODUCTION

Software agents have become a powerful technology to help on the development of distributed complex applications. Several methodologies have been proposed in order to allow the development of Multi-agent Systems (MASs). However, most of them do not take into account the adoption of extensive reuse practices that can bring an increased productivity and quality to the software development.

Software product lines (SPLs) [1] have emerged as a new trend of software reuse investigating methods and techniques in order to build and customize families of applications through a systematic method. Recent research [7, 3] has explored the integration between SPL and MAS. The aim of these new approaches is to integrate SPL and agent-oriented techniques by incorporating their respective benefits and helping the industrial exploitation of agent technology. Nevertheless, these approaches use a SPL perspective for particular purposes and do not address the development of SPLs to derive MAS [6].

In this context, this work presents an approach for modeling MAS-PLs, focusing at the domain analysis stage. We propose an approach that extends PASSI [2], an agent-oriented methodology, to support the management of SPL variabilities. PASSI provides a useful way for specifying a MAS, although it considers the development of single systems.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SAC'09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.
Copyright 2009 ACM 978-1-60558-166-8/09/03 ...$5.00.

2. APPROACH OVERVIEW

In this section, we present our approach for modeling MAS-PLs, based on the PASSI methodology. Due to the lack of expressivity for documenting variability, we propose extensions to document agency features.

PASSI (Process for Agent Societies Specification and Implementation) [2] is an agent-oriented methodology that specifies five models with their respective phases for developing MASs. The domain analysis stage corresponds to the System Requirements Model, which generates a model of the system requirements in terms of agency and purpose. PASSI methodology is designed for developing single systems, therefore we had to adapt it to express variability.

2.1 Feature Modeling

Feature modeling is an important activity in SPLs. It is the activity of modeling the common and variable properties of concepts and their interdependencies. The features are organized into a coherent model referred to as a feature model, which specifies the features of a product line as a tree, indicating mandatory, optional and alternative features. Besides the feature model, constraints express the feature interdependencies.

2.2 Domain Requirements Description

In this phase, we make a functional description of the system composed of a hierarchical series of use case diagrams. In order to enable the variabilities modeling, we have adapted these PASSI diagrams using the PLUS [4] method notation. In the PLUS approach, stereotypes are used to indicate if a use case is mandatory (kernel), alternative or optional.

Besides the stereotypes, we also colored the use cases to indicate which feature they are related to. This indication is

used in all artifacts. However, we also provide another view to the use cases to better express this relation: we grouped them into features with the UML package notation, as PLUS proposes. In addition, we have adopted the <<*agency feature*>> stereotype to indicate that the use cases of a specific package is related to an agency feature.

2.3 Agent Identification

The input of this phase are the use case diagrams generated in the Domain Requirements Description phase. Responsibilities are attributed to agents, which are represented as <<*agent*>> stereotyped UML packages. PASSI methodology considers that all functionalities of the system are performed by agents. Agents are entities that usually presents autonomy and pro-activeness, so only the functionalities that have these characteristics need to be performed by agents. Thus, the use cases that are into a UML package stereotyped with <<*agency feature*>> will be considered to be performed by an agent.

2.4 Role Identification

In this phase, all the possible paths, a scenario of interacting agents working to achieve a required behavior of the system, of the Agents Identification diagram are explored. The agent interactions are expressed through sequence diagrams. Usually, each scenario corresponds to only one feature of the system. For these cases, a feature dependency table to map sequence diagrams to each feature is enough. However, there are some features that impact another feature. We say that the feature crosscuts the other. The solution that we found for this problem is the use of UML 2.0 frames to express optional and alternative paths.

2.5 Task Specification

In this phase, activity diagrams are used to specify the capabilities of each agent. In these diagrams, we have made three adaptations, some of them were already adopted in other diagrams: (i) instead of drawing only one diagram per agent, we split the diagram according to the features; (ii) use of UML 2.0 frames to show different paths when there is a crosscutting feature; (iii) a colored indication showing with which feature the task is related to. The main objective of these adaptations is to provide a better feature modularization and traceability. Splitting the diagram in the way we proposed allow the selection of the necessary diagrams during the application engineering according to selected features.

3. DISCUSSIONS

In this section, we present and discuss some lessons learned while modeling agency features in MAS-PLs and challenges that we still have to face.

Integration of SPL techniques with existing Multi-agent Methodologies. Several MAS methodologies have been proposed, whose purpose is building agent-based systems; however each has its own unique perspective and approach to developing MASs. PASSI was chosen as the base of our approach because it integrates concepts from object-oriented software engineering and artificial intelligence approaches. It uses an UML-based notation, facilitating the incorporation of notations already proposed for SPLs [4].

Explicit Separation of the Modeling and Implementation of MAS Features from other Technologies. MAS methodologies usually propose to distribute all the system functionalities among agents. We claim that SPL features that do not take advantage from agent technology can be modeled and implemented using typical programming techniques. In our approach, we adopted the <<*agency feature*>> stereotype to indicate features that present autonomous or pro-active behavior and should be modeled using agent abstraction. This let us use the several technologies that exist for improving the development of web applications.

Crosscutting agency features. Crosscutting features have their design and implementation typically spread and tangled along different system modules. Our approach do not provide clear support to deal with the documentation of these crosscutting features, however we are currently investigating how existing aspect-oriented approaches can help the visual documentation of the agency features. We already have some results in this direction, by investigating agency features modularity [5].

4. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an approach for modeling MAS-PLs at the domain analysis stage. Our approach is based on PASSI methodology, which supported the specification of software agents. We have extended this methodology to address agency variabilities in SPLs. An important phase that needs to be added in the methodology is the feature modeling, which is the activity that identifies the SPL common and variable features. In addition, we have extended the PASSI notation, using stereotypes to indicate the variable abstractions and components of the systems. Since PASSI is based on the UML notation, it allowed us to adopt notations from PLUS, an existing SPL approach. We also discussed some important topics that arose from our study, such as the use of object-oriented techniques in agent-based applications and the need of a clear support for crosscutting features.

5. REFERENCES

[1] P. Clements and L. Northrop. *Software Product Lines: Practices and Patterns*. Addison-Wesley, USA, 2002.

[2] M. Cossentino. *From Requirements to Code with the PASSI Methodology*, chapter IV. Idea Group Inc., Hershey, PA, USA, 2005.

[3] J. Dehlinger and R. R. Lutz. A Product-Line Requirements Approach to Safe Reuse in Multi-Agent Systems. In *SELMAS 2005*, USA, 2005.

[4] H. Gomaa. *Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures*. Addison Wesley Longman Publishing Co., Inc., USA, 2004.

[5] C. Nunes, U. Kulesza, C. Sant'Anna, I. Nunes, and C. Lucena. On the modularity assessment of aspect-oriented multi-agent systems product lines: a quantitative study. In *SBCARS 2008*, Brazil, 2008.

[6] I. Nunes, U. Kulesza, C. Nunes, and C. Lucena. Documenting and modeling multi-agent systems product lines. In *SEKE 2008*, pages 745–751, Redwood City, USA, 2008.

[7] J. Pena, M. G. Hinchey, M. Resinas, R. Sterritt, and J. L. Rash. Designing and managing evolving systems using a MAS product line approach. *Science of Computer Programming*, 66(1):71–86, 2007.