The Method Engineering Process for Multi-agent System Development

Xue Xiao
Lab of Intelligent Science
Henan Polytechnic University
Jiaozuo,Henan, P.R.China
Email: jzxuexiao@126.com

Zhang XueYan
Department of Information Technology
Ningbo radio and tv university
Ningbo, Zhejiang,P.R.China
Email: zxueyan@gmail.com

Abstract

The agent-oriented (AO) methodology is an effective means for constructing distributed systems. Despite a great deal of research, a number of challenges still exist before making agent-based computing a widely accepted paradigm in software engineering practice. In order to solve the problem of "difficult to implement", the paper presents a method engineering development process for facilitating building up new agent system by reusing method meta models and design patterns. As the supporting tool, the hierarchical development architecture(HDA) plays a key role in the process. To exemplify its feasibility and effectiveness, the construction of C4I system is presented as a case study.

1. Introduction

Agents and multi-agent systems (MASs) offer a new and often more appropriate route to the development of complex computational systems, especially in open and dynamic environments. Therefore, in the last few years, there has been a great deal of research related to the identification and definition of suitable models and techniques to support the development of distributed systems in terms of MASs [1], such as formal modeling approaches, development methodologies and modeling techniques, specifically suited to the agent-oriented paradigm. However, despite the great deal of research in the area, agent oriented software engineering (AOSE) is still a relative young area. Currently, there are, as yet, not standard methodologies, development tools, or software architectures. There still exist a number of challenges before making agent-based computing a widely accepted paradigm in software engineering practice [2].

Because of high-level knowledge demand(such as AI, agent technology, AOSE and specific speciality), it is a challenge for most of developers to implement multi-agent system. At the same time, the rapid development of agent-oriented application begin to need different levels of quality in the design and its documentation. This brought us to identify the need for an reusable development process that could be supported by some design tool. Taking profit of our previous experience with related case study, we conceived a method engineering approach in the paper that allow agent developers to facilitate building up new agent system by reusing meta models and design patterns.

In order to support the process, a hierarchical development architecture(HDA) is proposed to combine different meta models originated from various AO methods to customize a best suited development methodology for the given project. Through applying the HDA-based design patterns, developers can build applications from third party off-the-shelf solution components, which bridge the gap between design abstraction and software implementation. The rest of the paper is organized as follows: Section 2 presents the theoretical background and related work. Section 3 introduces the reusable method engineering development process and the supporting tool(HDA) in detail. In section 4, a research project on the construction of C4I system exemplifies the effectiveness of the process as a case. Some conclusions are presented in section 5.

2. Theoretical Background

Some research works (e.g. [3]) provide comparison studies between different AO methodologies, showing that each AO method has its own weaknesses and strengths. Many users had trouble in finding a method that would satisfy their needs completely. Method engineering is a rational approach to the construction, either fully or partially, of methods from method fragments typically stored in a repository[4]. The method itself is constructed by selection of appropriate method fragments in such a way as to satisfy the requirements for the method and create a meaningful overall method. Ideally, a method engineering approach to method construction will utilize a process framework from which current and future method fragments can be gener-

e-Forensics 2008, January 21-23, 2008, Adelaide, Australia.

ated by the instantiation rule. Furthermore, a second set of rules and guidelines is needed to assist in method construction. We believe that the AO methods should be created and customized in a modular way, which enables developers to build project-specific methodologies from meta models, just like applications built from reusable off-the-shelf components.

As in other emerging technologies, it is common for agent developers to invent and re-invent repeatedly perhaps often not-so-smart solutions to recurrent problems. During the early work on MAS, developers recognized a number of recurrent patterns in the design of agent applications. These early patterns were found to be highly successfully for jump-starting users who were new to agent paradigm. This experience tells us that it is important to identify the elements of good and reusable designs for agent applications and to start formalizing people's experience with these designs. This is the role of design patterns. Currently, some researchers have begun to apply design patterns to MASs development. However, it is absent in guiding how to sort and apply all kinds of agent oriented design patterns, especially originated from different organizations or authors. To a large degree, whether the application of patterns is effective lies on developer's experience. With the number of the patterns increasing, it's becoming more and more difficult to select and apply appropriate design pattern.

3. The Proposed Approach

3.1. Two-layer Reuse Schema

In order to reduce the difficulty in developing agent application, our aim is to realize a two-level reuse mechanism: method reuse and design reuse. As shown in figure 1, method reuse lies on the method engineering process and design reuse focuses on design pattern-driven development.

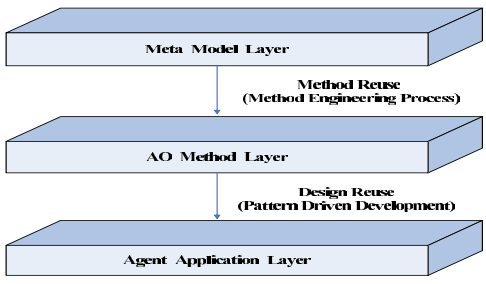

Figure 1. The Two-layer Reuse Schema

In the application of method engineering process, each successive pass will add additional detail and a series of system design artifacts is produced eventually, i.e. an ordered sequence of steps, an identifiable set of models, and an indication of the interrelationships between the models, showing how and when to exploit which models and abstractions in the development of a MAS. As fundamental element of constructing method, meta model plays a key role in the method engineering process. Developer can assemble a new methodology tailored to the given project by fitting meta models in appropriate position. Thus, each new project has its own methodology, built up from components of other methodologies. As a result, the advantages of different AO methods can be taken of and their drawbacks can be overcome.

The method engineering approach belongs to a "top-down" development policy, which focuses on design structure of the whole multi-agent system. Although very clear and well defined, it is not sufficient to meet all the requirements of implementing details of each model, which may lead developers without background in agent technology to be confused about how to turn design model into software implementation. What's more, this may result in re-inventing solutions to common design problems without benefiting from how they were resolved in the past.

For this reason we found useful to complement the method engineering development process with a kind of "bottom-up" development mechanism. A suited choice is to build an agent system incrementally from well-documented agent patterns, which can speed up development, avoids reinventing the wheel, and enables sufficient time to be devoted to the value added by the multi-agent paradigm.

3.2. The Method Engineering Process

In order to build our new design method based on reuse policy, we adopted(and extended) the method engineering paradigm. According to this approach, the new method is built by assembling pieces of method meta-models from a repository of methods, that is a structural representation of the method fragments that compose the actual AO method with their composing relationships. In this way we could obtain the best method for our specific needs. We chose this approach because, in the last years, it proved successful in developing many object-oriented applications, for example information systems, and is now collecting a growing interest from the agent community.

Figure 2 presents what we think to be the correct process for composing a new method under the evolution of the method engineering paradigm that we call agent-oriented method engineering. The process begins with the introduction in the method base of the fragments extracted from available OO methods and the specially created AO ones; then the designer(or better the method engineer), before building the new method, identifies the method meta models composing the kind of MAS he wants to build. Once the new method has been composed, it will be used by the designer when designing a system to solve some specific

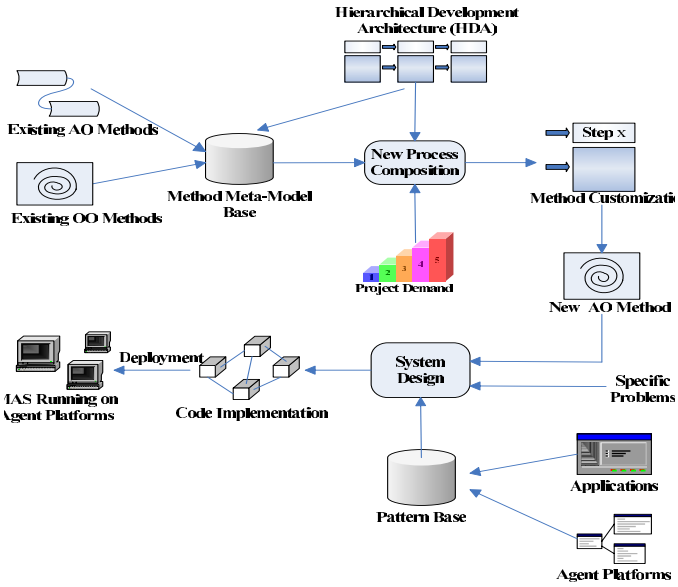

Figure 2. The adopted AO Method Engineering Process

problems. During the phases of detail design, agent oriented design patterns are introduced to support code implementation of MASs.

The composition of the new method is performed under the assistance of some specific design tool, called HDA(Hierarchical Development Architecture). In this process, the definition of the HDA will help at both a logical and practical level. Firstly, this will be useful in the method fragment selection phase(avoiding the selection of method meta models that are not dealing with the system to be built); secondly, offers a specific support for defining/composing our method fragments; and thirdly, the same fact of clearly declaring the structure of the system will allow to check for model coherence and to find not completely defined parts.

In creating a new method, we consider that this essentially is a design activity by itself and as such it should be ruled by some kind of design process. The design process we adopt(to design a new method) is composed of four phases: requirements analysis, process model design, fragments selection, and fragments integration(it includes the assembly and adjustment activities performed to adapt the fragments to the new process). Further iterations in this sequence of phases should aim at process maturity as described in the CMM but these aspects are out of the scope of this paper. The details are described as follows:

(1) The Requirements Analysis phase, consists of the identification of the important features of the method under consideration. Factor the overall problem and identify what layers should be selected in HDA for a specific application. Based on this, developers can extract appropriate meta models.

(2) The Process Model Design consists of the selection of the process model(waterfall, transformation,spiral,...), the phases that constitute it and other process level requisite (for instance the conditions that enable each new iteration). We consider situational requirements as the most useful guidance for selecting the right process model. The need of rapidly facing changing requirements could bring to the adoption of an evolutionary process while, conversely the need of a very formal development process could lead to the adoption of a waterfall model.

(3) The fragments Selection phase aims at identifying the best fragments for achieving the process goals(according to the requirements identified in the first phase).

(4) During the fragments Integration phase, the selected fragments are disposed in the right position inside the process and when necessary they are adapted to the new context. Method fragments contracts(preconditions required by each fragment and post-conditions enacted by it) are used to verify the possibility of directly connecting some fragments.

3.3. The introduction of HDA

Ensuring successfully use of the process will require seeking to solve those problems from a theoretical viewpoint. Against the background, the Hierarchical Development Architecture(HDA) is used to present as a means to unify concepts and promote reusability of systems to accommodate the application complexity.

As shown in figure 3, the HDA consists of five phases, covering the whole development lifecycle to establish a systematic development method. The requirement analysis phase defines development goal, which lays foundation for the following development. The MAS architecture phase represents the outline of the system configuration and organizational behaviors and is not dependent on any specific agent platforms. The agent modeling phase depicts all components of each agent specialized in agent structure. The software implementation phase gives the detail of system configuration and codes implementation according to the design of the above phases. Finally, the verification phase is used to ensure that the software to be constructed can meet the demand of users.

In the architecture, each phase is categorized into a layered model structure further. Based on the goal and the task of each layer, user can fill different meta models into the appropriate layers to handle different kinds of quality attributes. Instead of creating incompatible techniques, models and CASE tools for each methodology, modular and reusable meta models can be created once, and shared within different methodologies. It represents significant

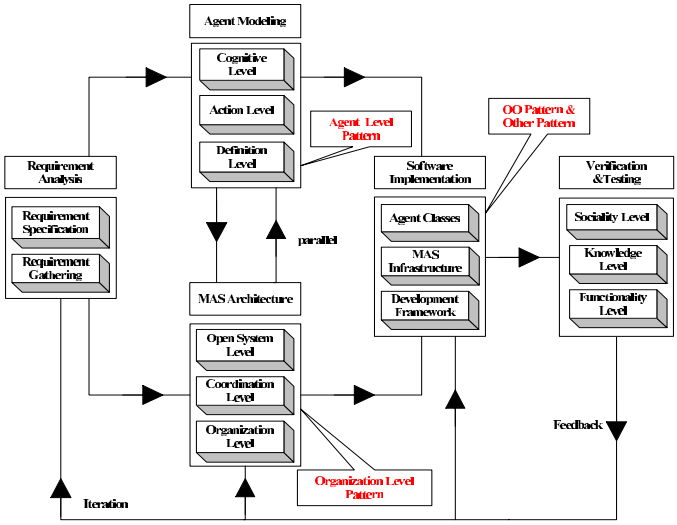

Figure 3. The schema of HDA

savings in development cost and learning cost. Developers can realize method customization through applying HDA in the following steps:

(1) Fill corresponding meta models into appropriate layers and form a new AO method through tailoring on particular project. The meta model which belongs to some layer should facilitate achieving the goal of the layer. Otherwise, it's not appropriate to put it into the layer.

(2) In order to meet the demand of customizing a new approach, developers need to enforce and modify the extracted meta models.

(3) The increasing details of a new approach to be constructed are developed step by step. Together, the layers enable the configuration of a new AO approach that can be appropriate for specific application.

(4) Once the method is composed, designers will perform the established process obtaining a model of the system - an instantiation of the new method - that solves their problem.

As a platform for sorting and managing design patterns, the HDA gives an explicit guide about how to apply those patterns and an order of examination of pattern application. As shown in figure 3, according to the position of the design pattern in the HDA, the patterns are classified into four categories: Organization Level Pattern, Agent Level Pattern, Object-Oriented Pattern and Others.

4. Application of HDA

The C4I (command, control, communication, computer and Information) system is the core of the whole naval warship, which is used as information process (including collection, transformation, process and transmission), fighting support and weapon control. C4I system integrates different kinds of weapons together to ensure that the integral performance can be improved to a high degree. Apart from the functionality, the C4I system must also satisfy properties such as reliability, fault-tolerance, scalability etc.

Aimed at those characteristics, we apply agent technology as an original and more effective way to solve highly-complex problems in the construction of C4I system. Through analysis and verification over and again, we select meta models from several representative AO methodologies to compose in HDA: RoadMap, Gia, MaSE, and Xiaoyuan's fish model. Through applying the method, a large-scale complex C4I system is decomposed and reconstructed successfully. The experimental results are satisfactory: on the one hand, the system meets the functional and non-functional demands(robust, reliable, scalable and so on); on the other hand, development efficiency have been improved greatly than before at a low cost. Because of limited space, we will have to diffusely discuss the details about customization in other papers.

5. Conclusions

The definition of agent-specific methodologies is definitely one of the most explored topics in AOSE. Because of the defects of current AO methodologies, it's difficult to turn AO software abstractions into practical tools for facing the complexity of constructing distributed systems. In order to improve the efficiency in MAS development, the paper presents a method engineering development process based on two-level reuse policy: method reuse and pattern resue. Through applying the HDA-based design patterns, developers can build applications from third party off-the-shelf solution components conveniently. In the case study of constructing C4I system, an agent-oriented new approach derived from the process is applied and the experimental result is satisfactory.

References

[1] M.Gervais, J.Gomez and G.Weiss. A survey on agent-oriented software engineering researches. *Methodologies and Software Engineering for Agent Systems, Kluwer: NewYork(NY)* (2004).

[2] F.Zambonelli, A.Omicini. Challenges and research directions in agent-oriented software engineering. *Autonomous Agents and Multi-Agent System* 253-283(2004).

[3] Khanh Hoa Dam and Michael Winikoff. Comparing Agent-Oriented Methodologies. In Proc of 5nd International Workshop on Agent Oriented Software Engineering (2003).

[4] B.Henderson-Sellers, C.Gonzalez-Perez, M.K.Serour. Method Engineering and COTS Evaluation. In Proceedings of the ICSE 2005, ACM Press:Montreal(2005).