Abstract

This paper introduces the new changes of contemporary enterprises and the supporting technology of Agile Management Information System. And then, this paper puts forwards the MAS-based enterprises structure because the Multi-agent system (MAS) is similar to the features of Agile Management Information System (AMIS) that is distributed, heterogeneous, re-configurable, reuse and scale to satisfy the needs of modern enterprises. It also points out that the complexity of AMIS, environment, and the interaction between them and suggests that modeling and simulation represent a major research direction in the area of AMIS. The paper presents the methods that allow a systematic and convenient simulation and modeling of AMIS to aim at firmly rooting simulation within the emerging agent-oriented software engineering. At last, it expounds above-mentioned views through an example.

Keywords: Agile Management Information System, Multi-agent System, Modeling, Simulation

1. Introduction

With the development and popularization of the Internet and computer, there have been dramatic social, political economic and technological changes, which have had a great impact on enterprises and forced them to change their structure, functionality and business strategies (Timmers 1999). Such changes have made enterprises adapt to new business environments (e.g. global markets), and adopt advanced technologies to support new and innovative ways of business (e.g. e-business). In order to adapt to this situation, people put forward MAS-based AMIS that has strict demands for the share-ability, acquirability, cooperation and consistency of information to really achieve the goal of sharing resources and guarantee for enterprises to reach their strategy goals. As a special information system, AMIS has its own unique property. It lays a strong emphasis upon flexibility, dynamic structure and efficient execution to support modern enterprises that is of multi-layer, cross-function, geographic-crossing and cross-boundary properties. However, in many cases, problems emerging in the AMIS are not efficiently reflected in the development phase. The development of AMIS not only faces problems associated with traditional distributed concurrent systems, but also encounters additional difficulties that arise from complex interactions between autonomous problem solving components. So, AMIS must be simulated and evaluated before being deployed. However, autonomous agents and open heterogeneous environment make simulation and evaluation more difficult than in the case of conventional software systems. Thus, the simulation of AMIS might become an important work. Although the number of simulation studies that analyze the temporal behavior of multi-agent systems in dynamic environments is increasing, the number of simulation systems is few. Among them hardly any exploits the art of modeling and simulation technologies (Uhrmacher 2002). According to the properties of AMIS, in this article we try to explain the system architecture and modeling method of AMIS, and to carry out and recur the design by layers respectively.
in accordance with the model of enterprise business. On the basis of what is said above, using Swarm we analyze the functions and performances of the system in order to decrease the risks in the system development and provide powerful guarantee for follow-up work.

2 Modeling OF AMIS

2.1 Structure Of Modern Enterprise
The modern enterprise is of multi-layer, cross-function, geographic-crossing and cross-boundary properties. The dividing lines between time and space become more and more blurred. Its reconfigurable organization structure has the characters of self-similarity, self-organization and self-optimization. At the top layer, there are the core enterprises and the cooperators, while the cooperators maybe are composed of sub-core enterprise and sub-cooperators, so on and so forth. Now we can describe the structure with the MAS (Figure 1), which can be viewed as an agent organization. In fact, a MAS evolves in a certain environment, has goals to achieve and its agents operate together to achieve these goals. For the more, modern enterprises not only emphasize the parties' separation but also pay attention to their coordination in order to achieve the strategic objects. So we feel that enterprise organization structure design should be somewhat inspired from MAS. The enterprise is made up of core enterprise agent and cooperator agents while cooperators maybe have the same structure.

![Figure 1. The enterprise organization structure](image_url)

2.2 Modeling of AMIS
AMIS has to support modern enterprise organization structure, so it must have the characters of openness, robustness, adaptability and effectiveness to provide following functions (Xu Qingsong et al. 2000).

1) Use open and adaptive architecture to support rapidly and dynamically reconfiguration of the organization structure. AMIS must provide a flexible and open architecture to satisfy the needs dynamically.

2) Reflect cooperators individual policies and remain them autonomous.

3) Accommodate cooperator heterogeneity. AMIS is usually composed of various hardware, platforms and so on. To achieve the information sharing, AMIS must accommodate cooperators’ heterogeneity.

4) Support the inter-operation across cooperators. It is very important to keep the data consistent to avoid faults.

5) Insure the security of the system. To insure the information not to be accessed by others with evil aims, some necessary measures must be provided in AMIS.

In modern enterprise it is the important way to communicate over Internet/Intranet. So the structure of AMIS is given in Figure 2. As shown in Figure 2, AMIS is composed of autonomous units with the networklization flat structure. The units with heterogeneous...
hardware, platform, and operation system are connected by the networks, which provide channels for collaborations through the special protocol. Every unit has the same components, such as cooperation agent, security agent, and application system. Cooperative agent is responsible for communication between the other units and the application system to fulfill the cooperation, while the security agent is in charge of insuring the inner information not to be accessed by others.

![Figure 2. The structure of AMIS](image)

The units are the basic components of AMIS. They have different functions and models, but we can describe them as MAS, which have the same structure (shown in Figure 3), in the high level with the agent-oriented modeling view. The objective of AMIS structure is to provide a universal architecture to describe the components and relationships of every business. The units must have following features.

![Figure 3. The structure of AMIS unit](image)

1) The units can be freely added in and removed according to the change of the business flow.
2) The relationships of different units are loose. On the contrary, the relationships between the components of the same unit are close.
3) The unit can perform autonomously.

In conclusion, we must take the following steps to construct the architecture of AMIS. First, we must formally represent the flexible enterprise business process with the workflow
perspective. Second, we must identify the process as different agents that can be integrated into some simulation platform in order to explore the behavior, analyze the function and performance of the alternative structure to provide the guidance to the designers and the decision-makers. At last, we must comply with the constraint principle of agent identification based on the boundary theory of the nature, which can be applied in agent identification. Through the analysis of openness and boundary, we can get hold of the main factors that influence the openness and the autonomy of agents to keep them autonomous, adaptive and cooperative effectively.

3. Simulation of AMIS

AMIS is a complex system (i.e. MAS) in essence. The best way of analyzing MAS that exhibits emergent phenomena or generates unforeseen patterns of spatial aggregation or global behavior is modeling and simulation. The simulation platform should test MAS in the small and in the large (Hanks et al. 1993). It can be used to reveal basic limits, problems, strengths, functions and performance of the proposed architecture. According to the results we can specify, develop, and maintain the architecture to evaluate the potential of the technology and the paradigm and to overcome the lack of quantitative results (Papaioannou 2000). However, developing, designing and finally implementing MAS is hard. There is neither a unified formal framework for MAS, nor a widely accepted methodology for development. Jennings and Wooldridge conclude, “systematic testing is the least developed area in developing multi-agent systems” (Jennings et al. 1998). Unlike many other scientific disciplines, systematic testing seems surprisingly unexplored in computer science even though testing plays an obvious and central role when selecting and assessing theories, methods and tools, and gaining new insights (Tichy 1998). The more complex programs become, the more often they are objects of empirical studies “because it is no longer true that we can predict how a system will behave by looking at its code” as Paul Cohen addresses in the preface to Empirical Methods for Artificial Intelligence (Praehofer et al. 1999).

Agent-oriented Software Engineering constitutes a young but growing branch of Software Engineering (Wooldridge et al. 2001) and research efforts are directed to structure and support an effective development of agent systems. Decomposition, abstraction, and organization are the basic strategies in designing multi-agent systems, as Jennings and Wooldridge dissert (Jennings et al. 1999; Jennings et al. 2003). That “agent systems work largely by emergent behavior and handle errors gracefully” (Petrie 2001) is another reason, which asks for simulation as an evaluation method employed during the agent development cycle.

Compared to traditional simulation methods, such as mathematical equations, discrete event-simulation, cellular automata, the simulation method we are in great need of is less abstract and closer to reality, since it explicitly attempts to simulate the actions of agent. There are many different formalization and implemented systems that are agent-based simulation. One of the most prominent is SWARM (Miner 1996). Swarm is a software platform for MAS simulation of complex systems. Swarm is based on the object-oriented paradigm. Memory management and time management in multi-agent simulation is the key features, which allow researchers to describe agents’ behaviors, all while keeping an exact notion of time and concurrency in the world. The basic architecture of Swarm is the simulation of collections of concurrently interacting agents. With this architecture, we can implement a large variety of agent-based models. Swarm also makes it possible to compose or decompose hierarchies of agents. Following the Swarm documentation (Swarm home page), we introduce a general sketch (shown in Figure 4) about how to implement a simulation in the agent-based modeling field. The first step is describing the real world system (the physical system) that is studied and decomposing it into a set of parts and events. The second step is identifying the agent. The third step is abstracting and deriving the
modeling and architecture of every agent. The fourth step is organizing the agents into collaboration and simulating. The last step is adjusting on the base of the simulation results.

Figure 4. The general sketch to implement an agent-based simulation

4. Case Study

4.1 Modeling Of Case
Consider an example of AMIS of the large-scale corporation founded 50 years ago. Now the corporation has a multi-layer, geographic-crossing and cross-boundary organization structure. To implement the system perfectly, we described and decomposed the whole workflow of the corporation, identified the agents of the AMIS, abstracted and derived the model of agents, and then constructed the architecture of the AMIS (shown in Figure 2). According to the simulation results, we analyzed and evaluated the functions and performances of the model to adjust the architecture we constructed. For the sake of simplicity, we just take the system of production unit as an example. It has the combination model of traditional C/S and B/S, supporting remote, multi-layer ordering management, producing management, stocking management, selling management and delivering management. After the steps mentioned above, the MAS-based structure of the system is described in Figure 5.

Figure 5. The MAS-based structure and agent interactions of production system

4.2 Simulation Of Case
Figure 6 describes the simulation implementation on the Swarm platform. There are two
kinds of swarm, the model Swarm and the GUI Swarm. The model Swarm is composed of agents included in the production system and the external systems. The agents hold all kinds of information, which are accessible by internal agents and other external agents. The agents perform certain functions to exchange information between agents. There is a shuffler mechanism to exchange the information among the agents. At every simulation step, all the agents are randomly matched. Initially, all the agents have their parameters, but in the course of the simulation, they evolve on an individual basis. The result is the data that reflect the functions and performance of the system, such as transaction efficiency, transaction capacity, and profit capacity. The GUI Swarm is used to display the results in different ways, such as graph, Histogram and so on. In the example, the resources utilization, network throughputs and waiting time as the simulation results are shown in figure 7.

Figure 6. The simulation implementation of production system in Swarm

Figure 7. The simulation results of AMIS
The Resource Utilization describes the utilization of resources during the system running. The Network Throughputs is the number of transactions in a second, which is influenced by the transaction capacity, the operation complexity, and the capability of operation system and database. The Response time is the time that a client receives the correct response after it sends a request. This index is the most important that users pay more attention to. In the simulation, we pay close attention to crucial factors while neglect some less important ones. As the trends shown in figure 7, we can draw the conclusion that the indexes of system performance satisfy the users after some times adjustment to the model.
5. Conclusion

In this paper, we try to set forth modeling and simulation method for MAS-based AMIS that allow a systematic, convenient, efficient and effective simulation and analysis and to firmly root simulation within agent-oriented software engineering. This paper has shown a method to achieve the goal. In the future simulation methods will play an important role in the area of designing AMIS. But the challenges such as multi-formalism modeling, variable structure models, interoperation of simulation systems, simulation and emulation, and parallel and distributed simulation are inevitable, we must be ready to overcome them.

6. References