Key Components of Agent-based Development

Amir Zeid

Computer Science Department, The American University in Cairo
azeid@aucegypt.edu

Abstract. In recent years, agent-based systems have received considerable attention in both academia and industry. The agent-oriented paradigm can be considered a natural extension to the object-oriented (OO) paradigm. In spite of the fact that there are many OO analysis and design methods, there is very little work reported on design and analysis of agent-based systems. Agents differ from objects in many issues which requires special modeling elements. In this paper, we propose a software engineering process for agent-based systems based on existing object-oriented software engineering concepts. We extend the Unified Modeling Language (UML) to model agents (static and mobile).

1.0 A UML profile for agent-based development

After a thorough investigation of OO analysis and design techniques it was found that they are not directly applicable to the development of multi-agent systems. This is basically due to the conceptual differences between objects and agents. After studying UML (Unified Modeling Language) which is the standard object-oriented modeling language, we found that mobile agents modeling is not supported by the current version of UML. The main goal then was to extend UML to model agent-based system development. The extensions include new modeling notations and concepts for both static and mobile agents. Our suggested agent-based process includes the following steps:

- Analysis

Analysis aims to clarify what a system is supposed to do, it mainly aims at answering "what" the system does rather than "how" to do it. The first step is to define the use-cases (expected tasks) of the system. The second step is to identify "places", agents and classes. The concept of places is associated with mobile agents. When a mobile agent transfers itself, it travels between execution environments called *places*. A *place* is a context within an agent system in which an agent can execute. Then, a static model of the identified agents, classes and places is constructed. The analysis process is concluded by filling a data dictionary.

- Design

Design deals mainly with how to implement the system to produce the expected outcomes as stated in the analysis phase. Design starts by defining authorization rights for places. After that, state machines for agents, places and classes are constructed. Then, a life-cycle model is constructed for each major task. Related places are then grouped in neighborhoods. The next step is to describe the dynamic behavior of the

system by defining mobile agents itineraries and modified interaction diagrams (extension to UML's interaction diagrams). The design phase is concluded by detailed design for methods (operations) and attributes of classes, agents, and places. Finally, the data dictionary is updated.

- Stereotypes (extensions to UML)

Stereotyping is the formal method to extend UML. The following table contains some of the extensions we proposed to extend UML.

Table 1. Summary of UML extensions for agents

Name	Agent	Mobile Agent	Place	Neighborhood	Itinerary Graph	Life Cycle	Meet
Base UML Meta Model Element	Class	Agent	Class	Package	Package	Package	Operation

2.0 Extending Rational tools to comply with our profile

We used the proposed profile to extend the Rational tool. Extensions includes new diagrams, new notations to represent agents, places and neighborhood and new operations and connection types.

- Upgrading Rational Configuration files

To upgrade the Rational tool, The *Stereotype Configuration File* has to be edited. A stereotype configuration file defines a set of stereotypes, including the location of icon files as well as settings that affect the usage and display of the defined stereotypes. Typically, the extension of a stereotype configuration file is .ini, but any extension is allowed.

- The new Class diagram menu

Figure (1) gives a snapshot of the new class diagram menu after adding our new elements: Agent, Mobile Agent, Place, Neighborhood, Authorized relation. The Figure also shows an example of a static relation diagram. It shows static agents, mobile agents, places, neighborhood, and classes. The model describes a virtual mall that contains shop classes, each shop has an inventory that may contain items. The Personal Communication Agent (PCA) travels from the user's host (given it has authorization rights) to the Virtual mall. The PCA can communicate with buyers and sellers to get the best price of the item he is looking for.

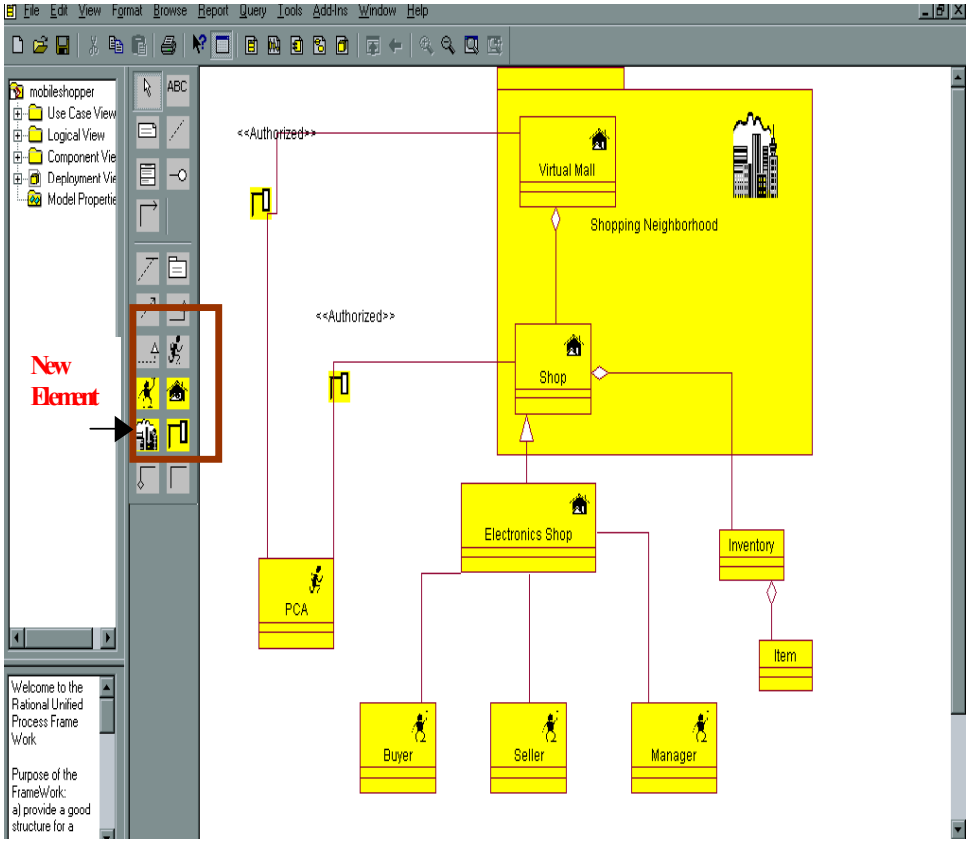

Fig. 1. Snapshot from Rational Tools with the new modeling elements

3.0 Conclusion

In this paper, we presented a new UML profile to model agents (static and mobile). The new profile includes notations for some new elements like agents, places and neighborhoods. The profile includes some new diagrams like the itinerary graph, authorization graph. The profile also includes some modifications to existing diagrams like the interaction diagram. We integrated our proposed profile with the leading Rational suite to provide a complete environment to model agents. Future enhancements will include code generation for agent-based languages.