

# AGENT-ORIENTED SOFTWARE MODELING

**Minjeong Kim, Seungyun Lee**

Department of Computer Science  
Sogang University  
Sinsu-Dong, Mapo-Gu, Seoul  
121-742, REP. of KOREA  
+82 2 701 4797

{mjkim,sylee}@selab.sogang.ac.kr

**Injae Park, Jintae Kim**

Department of Computer Science  
Sogang University  
Sinsu-Dong, Mapo-Gu, Seoul  
121-742, REP. of KOREA  
+82 2 701 4797

{jtkim,bkdraft}@selab.sogang.ac.kr

**Sooyong Park**

Department of Computer Science  
Sogang University  
Sinsu-Dong, Mapo-Gu, Seoul  
121-742, REP. of KOREA  
+82 2 705 8928

sypark@ccs.sogang.ac.kr

## ABSTRACT

Due to the increased applications of agents, Agent-oriented software becomes large and complex. To support systematic developments of such a software, Agent-oriented software development methodology needs to be developed. This paper focuses on modeling phase of agent-oriented software development. For the Agent-oriented software modeling, *Agent Elicitation method*, *Intra and Inter Agent modeling method* are proposed. *Agent Elicitation* shows how to extract agents from classes in the real world. Modeling methods of Agent's characteristics - Goal, Belief, Plan and Capability - are proposed for *Intra Agent modeling*. Methods of Agent's mobility and communication in multi-agent systems are proposed for *Inter Agent modeling*.

## Keywords

Agent Modeling, Requirement Analysis and specification, Agent Oriented Software Engineering (AOSE)

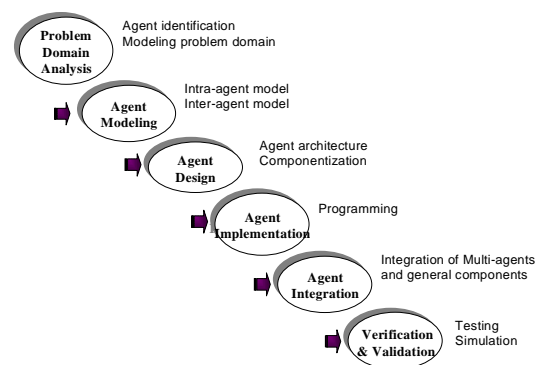
## 1 INTRODUCTION

As software applications are getting complex and large, various environments are needed to support heterogeneous and distributed real-time applications. For this reason, the agent has introduced as one of the solutions in future software environments. Since the agent had introduced from AI community, it has extended to various applications, such as e-mail filtering, Air-traffic Control and etc[1]. Moreover, in distributed and heterogeneous environments such as Electronic Commerce(EC) applications, the concepts of agent are widely applied.

As agents are used in many application areas, the systematic approach with software engineering paradigm is getting more important in agent-oriented software development. However, there has not been enough research in this area.

To develop agent-oriented software, we introduce process model depicted in [figure 1]. Based on this, our paper focuses on the agent modeling that include agent identification, problem domain modeling, and Intra and Inter Agent modeling method.

We believe that the real world consists of agents and objects, and an agent is a kind of Active objects[1] or distributed objects[2]. Based on the proposed process model, we obtain objects from the problem domain analysis using UML(Unified Modeling Language)[3,4], then, generate and extract agents from objects using *agent selection rules*. In



**[Figure 1] Agent-Oriented Software Development Process Model**

the Agent Oriented modeling, the system is divided by two parts; *Intra agent model* and *Inter agent model*. The former shows agent's attributes and behaviors, and the latter does message exchange and agent's mobility.

In Section 3.1, UML based problem domain analysis is explained. Agent Elicitation process based on *Agent Selection Rule* and *Agent-Class Diagram* are described in Section 3.2. In Section 3.3 and 3.4, *Intra Agent modeling* - goal, belief, plan and capability, *Inter Agent modeling* - mobility and communication, are proposed. Finally, we applied our approach to an Electronic Commerce domain to validate our modeling method.

## 2 AGENT'S PROPERTIES AND AGENT MODELING METHODS

The concept of agent was started from John McCarthy in the mid-1950's and established by Oliver G. Selfridge several years later. In the early days, many researchers had

been studied about agent in boundary of AI. Since 80's the agent had become widely spreaded and applied.

Agents are assumed to have following three properties[6].

- **Autonomy** – agent can make decisions about what to do based on its own states, without the direct intervention of human or others.
- **Adaptation** – agent can perceive its environment, and respond to its changes in a timely fashion.
- **Cooperation** – agent can interact with other agents via some kind of agent-communication language, and typically has the ability to engage in social activities to achieve its goal.

In agent modeling, it has been continued to study on the conceptualization of complex systems, that consists of Agents. Rao-Georgeff's BDI(Belief-Desire-Intention) model[7] is regarded as one of the relatively successful Agent model. Based on previous works, Kinney proposed the design methodology to develop MAS(Multi Agent System) extending OMT[13]. He proposed two main views of BDI agent modeling; internal and external view with Abstraction, Inheritance and Modularity. Burmeister[14] describes an Agent-Oriented(AO) methodology, extending OO techniques by agent model, organizational model and cooperational model. Recently, Falchuk and Karmouch proposed the modeling method about mobility and communication of the agent called Visual agent modeling[15]. They defined various elements of agent's active environment, such as agent, server, data, document, etc., with 26 icons. Introducing the Iconic Modeling Tool(IMT), they tried to help the user visualize and model mobile agents and their itineraries. Although they proposed visual modeling methods for the first time, internal aspects of agent are omitted.

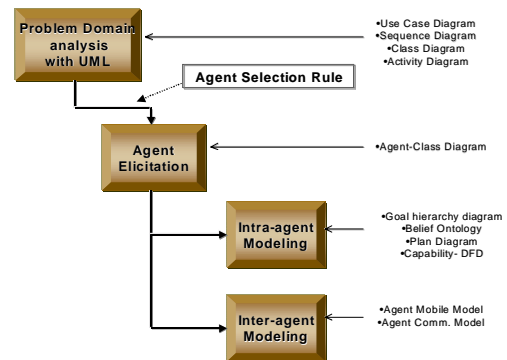
However, there is the lack of definite classification between agents and objects in the system analysis procedure. To model static agents and mobile agents in multi agent environments, the unified modeling method is needed. Therefore, our research will suggest a new agent oriented modeling methodology to provide the integrated and unified modeling method.

### 3 AGENT -ORIENTED MODELING METHOD

Proposing Agent-Oriented approach to model the real world, we assume that the real world has following characteristics:

- Objects and agents are co-existed and they have mutual relationships.
- An active object can be regarded as an agent[1].
- Agent acts asynchronously.
- Interactions among agents are took place by message exchanging.

Based on above assumptions, Agent-Oriented modeling process is proposed as depicted in [Figure 2]. This process



[Figure 2] Agent Oriented Modeling Process

can be divided into 4 steps; UML based Problem Domain

Analysis, Agent Elicitation, Intra Agent Modeling, and Inter Agent Modeling. UML based Object-Oriented Analysis Method is used for the problem domain analysis. From the validation of its utility in industry fields, the UML approach can give an objective view in the early stage of Agent Elicitation. It also provides deep understanding of the problem domain with static and dynamic aspects of the system. After analyzing the problem domain, objects that can be agentified and agents that need to be added are determined by agent selection rules in [Tabel 1]. These objects and agents can be expressed in Agent-class Diagram. Agent's internal characteristics are proposed in intra agent modeling. Mobile agent model and agent communication model are shown in inter agent model.

#### 3.1 Problem Domain Analysis with UML

For the problem domain analysis, UML approach is used. With UML[3,4], not only static but dynamic aspects of system can be analyzed and agent elicitation can be conducted. Diagrams used in problem domain analysis are represented below.

1. *UseCase Diagram* : A UseCase diagram is a diagram that shows a set of use cases and actors and their relationships. It supports the behavior of a system by modeling static aspects of a system.
2. *Sequence Diagram* : Sequence diagram shows an interaction, consisting of a set of objects and their relationships, emphasizing the time ordering of messages. It models dynamic aspects of a system.
3. *Class Diagram* : Class diagram shows a set of classes, interfaces, and collaborations and their relationships. It address the static design view of a system, showing a collection of declarative elements.
4. *Activity Diagram* : Activity diagram shows the flow from activity to activity. Activities ultimately result in some action, which is made up of executable atomic computations that result in a change in state of the

system or the return of a value. It is used to provide dynamic aspects of a system such as system functions.

### 3.2 Agent Elicitation

With the consideration of static and dynamic aspects of a system, objects that need to be agentified are determined by applying Agent Selection Rule. Each rule is closely related to the Agent's characteristics.

1. Autonomy
1.1 Does it need internal knowledge?
1.2 Does it make decision by itself?
1.3 Can it be tolerant of errorful, unexpected, or wrong inputs?
2. Adaptation
2.1 Does it need internal knowledge?
2.2 Is its knowledge updated continuously?
2.3 Does it interact with external entities?
3. Cooperation
3.1 Does it interact with external entities?
3.1.1 Does it operate cooperatively?
3.1.2 Is it operated in multi-threaded style?

Table 1 Agent Selection Rule

To represent the relationships among agent's attributes, we define meta-rules. Nwana[13] proposed that agents can be classified into 4 groups; smart agent, collaborative learning agent, collaborative agent, interface agent.

1. If the agent has autonomy, adaptation, and cooperation, it is called as "smart agent".
2. If the agent has adaptation and cooperation, it is called as "collaborative learning agent".
3. If the agent has autonomy and cooperation, it is called as "collaborative agent".
4. If the agent has autonomy and adaptation, it is called as "interface agent".

Table 2 Meta Rule

Elicited agents are classified as mobile agent and general agent(stationary agent), based on mobility of agent. Agents produced by agent selection rules, and objects remained in class diagram are expressed in Agent-Class diagram.

#### 3.2.1 Agent-Class Diagram

Based on the assumption that objects and agents exist together in the real world, Agent-class diagram shows the relationships among agents and objects in target problem domain. Additional elements are represented as follows:

#### Elements

- $\textcircled{A}$  Agent without mobility.
- $\textcircled{M}$  Mobile agent
- Class : Established classes

#### Relation

- Cooperation ( $\text{---}\bigcirc\text{---}$ ): shows the relationships among agents, meaning that cooperation is needed among them.
- Employ( $\text{---}\textcircled{E}\text{---}$ ): shows the relationships between agents and classes, meaning that agent can use established classes.

Agent-Class diagram expresses agent's internal characteristics - goal, belief, plan, and capability- as its attributes.

### 3.3 Intra-Agent Modeling

Intra agent modeling proposed in our research is based on BDI model[7] and Reticular Agent Mental Models[9] proposed by Thomas. We propose agent's 4 characteristics; goal, belief, plan, and capability. [Figure 3] shows an abstract view of Intra agent models.



[Figure 3 ] Abstract view of Intra agent

#### 3.3.1 Goal Model


Goal is an ultimate objective of an agent to be achieved. In KAOS[10]' approach, goals are identified in the problem domain analysis. It can be expressed as Pattern\_of\_Goal[objective] and pattern\_of\_Goal can be categorized into five groups; Achieve, Cease, Maintain, Avoid, and Optimize. Goal hierarchy diagram shows cooperation relations among agents corresponding to each goal, using Goal structure. Expressions used in Goal hierarchy diagram are represented below.

#### Element

- Nonfunctional Goal : an objective achieved by the entire system
- Operationalizable Goal : an functional objective achieved by agents

#### Relation

- AND( $\text{---}\bigwedge\text{---}$ ) : Every sub-goals should be satisfied to achieve the parent goal
- OR ( $\text{---}\bigvee\text{---}$ ): One or more sub-goal should be satisfied to achieve the parent goal

- Conflict (  ): shows whether one goal conflict with other goals

### 3.3.2 Belief Model

Belief is considered as data that agent already has. It is an information about environments and the agent itself. These data should be updated if necessary. They should be established as knowledge base, named Ontology which can be built by Ontolingua[11] or KIF(Knowledge Interchange Format)[12].

The rules that determine what kinds of ontology are needed are listed below.

#### 1. Ontology that can be established in the early stage

- Information about Protocol or ACL (Agent Communication Language) for the agent's communication can be established as Ontology in the early stage.

#### 2. Ontology that needs to update continuously

- Attributes and operations in Class diagram can be established as ontology.
- Messages among objects in Sequence diagram can be established as ontology.
- Agent's information that needs to be built as a knowledge base can be established as ontology

### 3.3.3 Plan Model

Plan shows agent's behavior to achieve a goal. It focuses on agent's behavior changes as time flows and shows messages exchanged among agents. Based on the analysis of dynamic aspects of systems, each agent can determine its own behavior, referencing its goal and belief. *Agent plan sequence diagram*, which extends established Sequence diagram, represents agent's behaviors. Additional elements are defined below.

- **Mobile(Goal[Objective])** : the agent with its goal tends to mobile to other domain
- **Update(Belief[Type\_of\_ontologies])** : the agent updates its ontology corresponding to its information
- **employ** : the agent uses classes
- **msg[message\_context]** : message that exchanged among agents

### 3.4 Capability Model

Capability is the operations that agent can perform. It explains about agent's internal changes from inputs and outputs. It can be represented by using DFD(Data Flow Diagram).



[Figure 4] General Form of Agent's Cap.

## 3.4 Inter-Agent Modeling

Agent's mobility and messages exchanging among agents in multi agent systems are modeled in *Intra agent modeling* process. This has 2 models; Agent Mobile Model and Agent Communication Model.

### 3.4.1 Agent Mobile Model

The Agent Mobile Model shows how the agent migrates to perform a task. It also shows mechanisms that determine the destination that the mobile agent migrates; the basis of mobile agent migration and which host the agent will move to. Since we cannot tell how the agent decides its destination from the external viewpoint, agent's internal view(the mental state) is needed to know the agent's decision mechanism.

### 3.4.2 Agent Communication Model

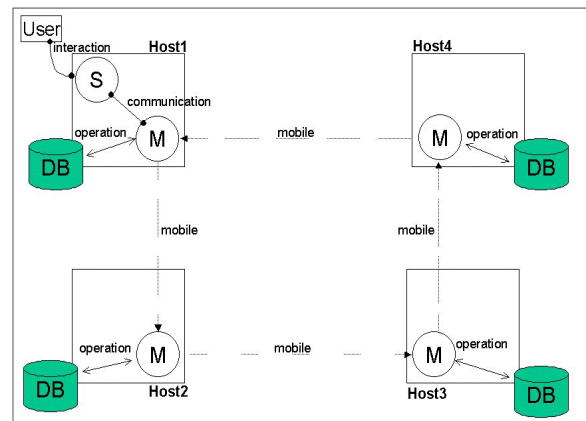
The Agent Communication Model shows messages exchanging among agents. It shows which agent communicates with other agents and which message format is used for their communication. It also shows which information is included in their messages and which process is needed to create those messages.

## 4 AN EXAMPLE ON ELECTRONIC COMMERCE DOMAIN

In this section, we apply our modeling approach to the electronic commerce domain. Start with the explanation of the problem domain and its scenario, we applied proposed Agent's modeling methods.

### 4.1 Descriptions about the problem domain and its scenario

As the reduced scale of EC domain, there are four servers connected to network. Two hosts are operating on Window 98, one on Window NT, and one on Solaris. Each host has the database contained the information of items, which are components of PC. Mobile agent helps to find the best information about the expected item. [Figure 5] is an abstract view of the whole system and the detail scenarios are described in [Table 2].



[Figure 5] System Diagram of Problem Domain

User A intends to purchase components of PC through the Internet. User A enters the name of the component, to Product Info Seeker Agent. The Agent migrates to search for the information of the item, which the user requires. Agent compares with each information about items. The Agent sends the information to the other Agent. The Agent asks User A to order that item. If User A wants to order, the Agent processes the requirements.

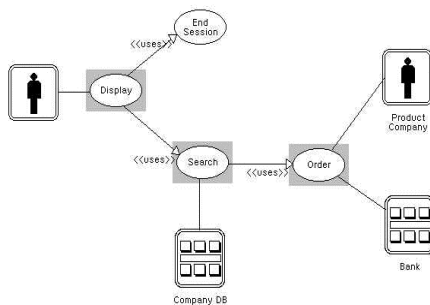
[Table 2] Our Example Domain Scenario

## 4.2 UML based Agent Elicitation

We apply our modeling approach to the EC problem domain analysis. UseCase Diagram, Sequence Diagram, Class Diagram, and Activity Diagram are produced and Agent Selection Rules are applied for the Agent Elicitation. Finally, we produce Agent-Class Diagram for our example domain.

### 4.2.1 UseCase Diagram

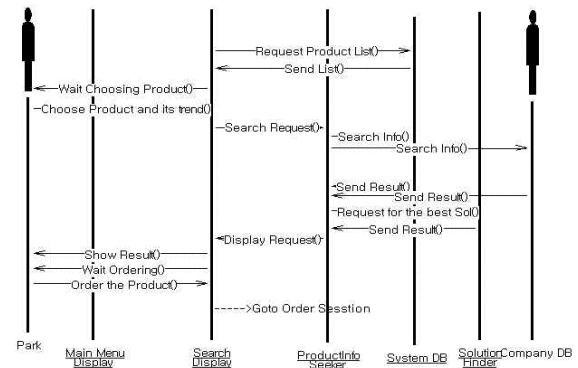
There are 4 actors in our example; the user who enters the information, the company which has the actual product, company's databases which have the information about the products, and the bank which can check the user validation. There also 3 usecases; Display usecase which takes charge of communication with customers, Search usecase which find the information about products, and Order usecase which processes the ordering. [Figure 6] shows the UseCase Diagram applied to our example domain.



[Figure 6] UseCase diagram applied to example domain

### 4.2.2 Sequence Diagram

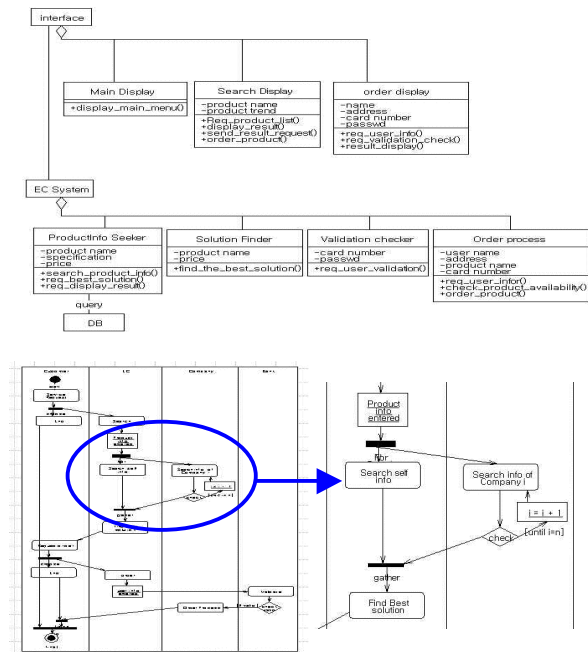
For each usecase, we produce Sequence diagram. [Figure 7] shows the Sequence Diagram about Search usecase, which performs the main tasks.



[Figure 7] Sequence Diagram applied to example

### 4.2.3 Class Diagram and Activity Diagram

Extracting objects from Sequence Diagram and Activity Diagram, Class and Activity diagram are produced. We represent attributes and operations of objects in static and dynamic aspects. [Figure 8] shows Class diagram and Activity diagram applied to our example domain.



[Figure 8] Class Diagram(top) and Activity

### 4.3 Agent Elicitation and Agent-Class Diagram

Applying Agent Selection Rules to the UML based diagrams, we determine objects that can be agentified or agents need to be added. If one or more classes reach to the agent, we represent them in one group. From Class diagram and Activity diagram, 3 Agents(Interface Agent, Product Info Seeker Agent, and Solution Finder Agent) are elicited and one new agent(Coordinator Agent) is produced. [Table 3] shows the rules applied to each agent.



**Interface Agent** : Communicating with the user, the Agent understands his/her intentions and exchanges messages with Product Info Seeker Agent to get the information. It has autonomy and adaptation as its attributes(Rule 1.1, 1.3, 2.3 are applied).

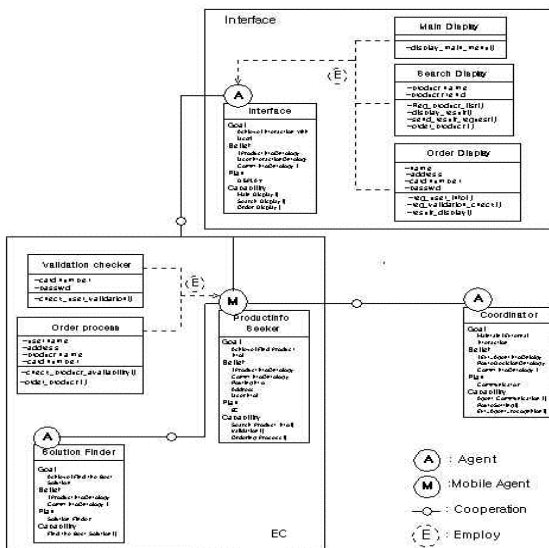
**Product Info Seeker Agent** : As performing the main role, the agent moves to databases and finds the information of the product, checks the user validation, and processes the ordering. This agent is applied to the distributed environments and it cooperates with other agents. It is the mobile, collaborative agent which has autonomy and cooperation(Rule 1.1, 3.1(3.1.1, 3.1.2) are applied).

**Coordinator Agent**: Perceiving the external environments, if there is any changes, the agent should update its internal knowledge and provides the route that informs the next mobile place to the Product Info Seeker agent. This is the newly created smart agent that helps to perform agent's mobility efficiently, having autonomy, adaptation and cooperation(Rule 1.2, 2.1, 2.2, 3.1.1 are applied).

**Solution Finder Agent** : From the information the Product Info Seeker Agent has found, Solution Finder Agent determines the best result. It is the collaborative agent which has autonomy and adaptation(Rule 1.1, 1.2, 3.1.1 are applied).

[Table 3 ] Elicited Agents based on Agent Selection Rules

[Figure 9] shows Agent-Class Diagram for our example domain. Product Info Seeker Agent is expressed as **M** to represent its mobile characteristics.



[Figure 9 ] Agent-Class Diagram applied to example domain

#### 4.4 Intra Agent Modeling

Goal, belief, plan, and capability are represented in Intra agent modeling process.

##### 4.4.1 Goal Modeling

Every agent has a goal and performs one or more tasks to achieve its goal. In our example domain, 4 agents have their own goal, as shown in [Table 4].

**Interface Agent** : Achieve[Interaction With User]

- It serves successful results to the user, interacting with him/her efficiently

**Product Info Seeker Agent** : Achieve[Find Product Info]

- It searches the product information successfully

**Solution Finder Agent** : Achieve[Find the Best Solution]

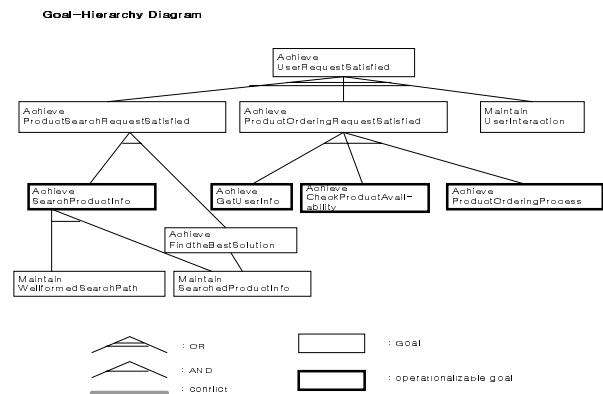
- It finds out the best one from the product information that Product Info Seeker Agent has found

**Coordinator Agent** : Maintain[External Interaction]

- It perceives external environment and maintains its information

[Table 4] Goal Modeling of Each Agent

To achieve the goal, each agent has several sub-goals that should be achieved in the first place. [Figure 10] shows Goal-hierarchy diagram, which shows the relationships among goals and their sub-goals.



[Figure 10 ] Goal Hierarchy Diagram applied to

#### 4.4.2 Belief Modeling

Following the ontology decision rules mentioned in the previous section, [Table 5] represented ontologies established for our example domain.

**ProductInfoOntology** : It is the overall knowledge for the user to choose the product. The product kind and its trend are established as ontology(Attributes of the Search Display class are applied).

**UserInteractionOntology** : To understand user's input about ordering process, user information such as name, address, credit card number, and etc. is established as ontology(Attributes of the Order Process class are applied).

**CommunicationOntology** : When agents need to cooperate one another, protocol or ACL(Agent Communication Language) are established as ontology(Requested operations of classes are applied).

**ExternalAgentInfoOntology** : The overall knowledge to perceive the external environments is established as ontology(Interacting messages in Sequence diagram are

applied).

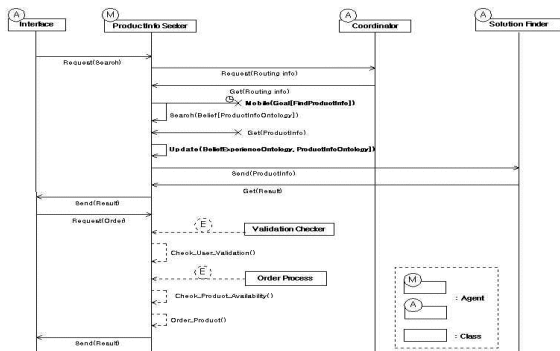
**RouteDecisionOntology** : The knowledge about the route, which needs for moving to other databases, is established as ontology(Interacting messages in sequence diagram are applied).

[Table 5] Ontology and its decision rules

Among ontologies mentioned above, Product Info Seeker Agent has ProductInfoOntology and CommunicationOntology. In addition, it has RoutingInfo, UserInfo and address as the basic belief information.

#### 4.4.3 Plan Modeling

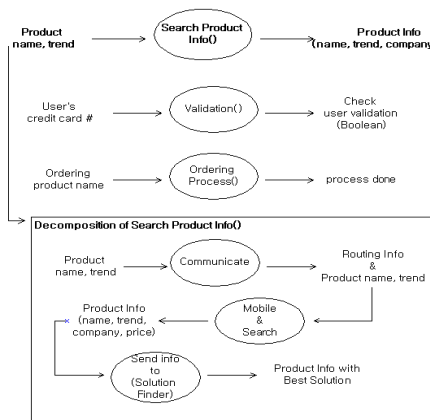
Plan represents agent's behavior to achieve the goal. [Figure 11] shows the plan characteristic of Product Info Seeker Agent, specifying messages among agents in Interaction Diagram.



[Figure 11] Plan Sequence Diagram of ProductInfoSeeker Agent

#### 4.4.4 Capability Modeling

Capability represents operations that are needed for achieving the goal. [Figure 12] shows capability of Product Info Seeker Agent, using DFD(Data Flow Diagram). The agent has 3 capability; Search Product Info(), validation(), and Ordering Process(). For more detail explanation of Search Product Info(), it can be decomposed into 3 sub-capabilities; Communication, Mobile and Search, and Send info to Solution Finder Agent.



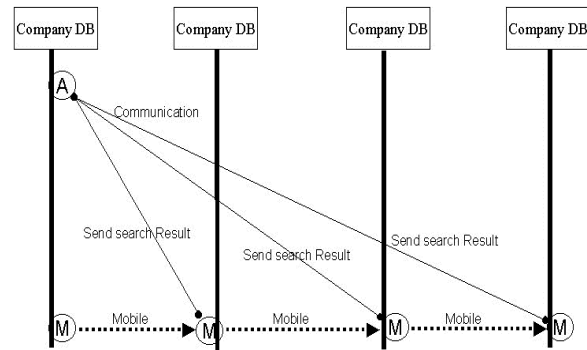
[Figure 12] Capability Model of ProductInfo Seeker Agent

## 4.5 Inter Agent Modeling

Agent's mobility and the messages exchanging among agents are represented in Inter agent modeling process.

### 4.5.1 Agent Mobile Modeling

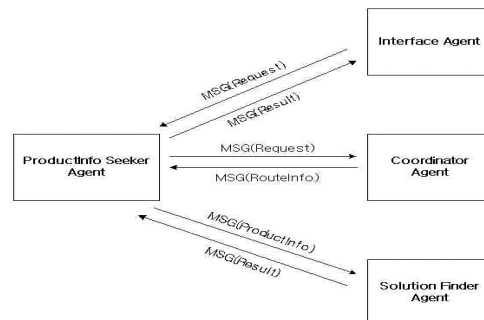
The Agent Mobile Model shows how the agent migrates to perform a task and mechanisms that determine the destination that the mobile agent migrates. [Figure 13] shows agent's mobile mechanisms applied to our example domain. Agent's internal view is shown abstractly in the diagram under bar in Figure 13. This is about agent's decision mechanism for the next destination. According to G1(the previous goal)and IOE(Information of Environment) as inputs, Decision Making State brings forth G2(the next goal) and next address as outputs. IOE has all information about agent's state. For instance, IOE informs you where Mobile Agent is, how much Mobile Agent processes its task. If G1 is equal to G2, the previous goal is the same as the next one and the given task is finished. If not, G2 is the subgoal of G1 and Mobile Agent should process G2(the next goal). Product Info Seeker Agent is considered as Mobile Agent. It sends the result message from databases, communicating with Coordinator Agent.



[Figure 13] Agent Mobile Model applied to problem domain

### 4.5.2 Agent Communication Modeling

The Agent Communication Model shows messages exchanging among agents. It shows which agent communicates with other agents and which message format is used for their communication. [Figure 14] shows Agent Communication model applied to our example domain.



[Figure 14] Agent Communication Model applied to problem domain

To generate messages, the result value from agent's operation and the given goal are entered as inputs. These inputs generate the messages for the communication by agent's internal processing routine. The Message consists of three parameters; sender\_address(sender's address), receiver\_address(receiver's address), and message content.

## 5 CONCLUSIONS AND RESEARCH ISSUES

In this paper, we proposed agent modeling method in the real world by using Agent Elicitation, Intra and Inter Agent Modeling. In identification of object, UML approach were used. We suggested criteria on how to select objects that should be agentified and produced Agent-Class Diagram, which described static parts of a system and cooperations among agents. In Intra Agent modeling, we proposed Agent's internal modeling. Goal-hierarchy diagram represented cooperation relationships among agents corresponding to each goal and belief modeling proposed classification criteria on whether the knowledge should be established as ontology or not. Plan Sequence Diagram described changes of Agent's behavior and DFD(Data Flow Diagram) represented Agent's Capability. In Inter Agent Modeling, we have modeled Agent mobility and communication among agents. We have tried our proposed modeling method by applying it to small electronic commerce example.

Ongoing work includes formal definition of proposed method as well as development of supporting tool. We regard our research as a start point in developing modeling methods for the entire environments. Improvement and application for modeling method to electronic commerce in real world for the model validation should be continued. We will extend our research to other phases of Agent Oriented Software Development Methodology as the systematic approach.

## REFERENCES

1. N. R. Jennings, K. P. Sycara, and M. Wooldridge A Roadmap of Agent Research and Development In Journal of Autonomous Agents and Multi-Agent Systems. 1(1), pages 7-36. July 1998.
2. M.Schroeder, Are Distributed objects agents?, *International Bi-Conference Workshop on AGENT-ORIENTED INFORMATION SYSTEMS (AOIS'99)*, 1999
3. M. Wooldridge. Agent-based Software Engineering. In *IEE Proceedings on Software Engineering*, 144(1), pages 26--37, February 1997.
4. RAO.A.S and GEORGEFF, M.P.: 'Modeling rational agents within a BDI-architecture'. *Proceedings of Knowledge representation and reasoning (KR&R-91)*, (Morgan Kaufmann Publishers, 1991), pp. 473-484
5. Thomas,S.R., PLACA, An Agent Oriented Programming Language, PhD Thesis, Stanford University, 1993
6. Dardenne, A. van Lamsweerde and S. Fickas, Goal-directed Requirements Acquisition, *Science of Computer Programming*, 20, pp.3-50, 1993.
7. T.R.Gruber, A Translation Approach to Portable Ontologies, *Knowledge Acquisition*, 5(2), 199-220, 1993.
8. UMBC Lab for Advanced Information Technology, KIF ( Knowledge Interchange Format), <http://www.cs.umbc.edu/kse/kif/>
9. David Kinny, Michael Georgeff, and Anand Rao. A methodology and modelling technique for systems of BDI agents. In W. van der Velde and J. Perram, editors, *Agents Breaking Away: Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World MAAMAW'96*, (LNAI Volume 1038). Springer-Verlag: Heidelberg, Germany, 1996.
10. Birgit Burmeister. Models and methodology for agent-oriented analysis and design. In K Fischer, editor, *Working Notes of the KI'96 Workshop on Agent-Oriented Programming and Distributed Systems*, 1996. DFKI Document D-96-06.
11. Benjamin Falchuk and Ahmed Karmouch, "Visual Modeling for Agent-Based Applications", *IEEE computer*, December 1998.
12. M.J.Kim, J.T.Kim, I.J.Park, S.Y.Lee, and S.Y.Park, Agent-based Software Analysis Method in Distributed Environment, *Fuzzy-IEEE'99*, September 1999.
13. Nwana, H. S. "Software Agents: An Overview", *Knowledge Engineering Review*, 11(3), pp. 205-244, 1996