NAIST-IS-DT9861004

Doctor's Thesis

Studies on Autonomous Agent's Architecture with Knowledge Migration in Social Agency Model

Noriko Etani

February 5, 2001

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

Doctor's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
DOCTOR of ENGINEERING

Noriko Etani

Thesis committee: Masatsugu Kidode, Professor
Hideaki Takeda, Associate Professor
Shunsuke Uemura, Professor
Katsumasa Watanabe, Professor

Studies on Autonomous Agent's Architecture with Knowledge Migration in Social Agency Model*

Noriko Etani

Abstract

In multi-agent environment, each agent can be working at common goals with globally cooperative behaviors. In order to form cooperation, agents first migrate related knowledge, locally evaluate the others' requirements, then agents finally can form a plan to achieve goals. In order to construct a model integrating agent's behavior and cooperation among agents, we propose four goals; (1)cooperation of software agents in the multiple processes of its architecture and the communication between agents to achieve a common goal, (2)adaptability of a software agent when its autonomous software agent can control its correct behavior in the environment and can manage both its knowledge and other agents' migrated knowledge to execute its behavior in knowledge-level, (3)mobility of a real-world mobile agent when mobile computer and autonomous mobile robot equipped with a network can execute its behavior by knowledge migration between mobile computer and autonomous mobile robot, and (4)transparency of knowledge migration when the communication requires to construct transparent knowledge boundaries between real space and virtual space which a computer generates in its display.

In this dissertation, we present two approaches for agent collaboration to resolve the above mentioned issues. As for the first approach, we introduce social agency model for constructing a prototype system for guide activities in a laboratory. We,then, formalize the interaction between agents based on the notion of rational agents. As for the second approach, we present an autonomous

*Doctor's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT9861004, February 5, 2001.

agent's architecture in social agency aimed at communicating with other agents in knowledge-level. Its architecture is designed by three criteria; (1)to execute services on heterogeneous environments, (2)to develop independent software components, and (3)to negotiate agent's behavior between software components using protocols.

The main contribution of this dissertation has been to introduce role models in object-oriented software engineering, propose the agent's model to determine both agent's behavior and cooperation among agents allowing to express (1)cooperation, (2)adaptability, (3)mobility, and (4)transparency, and verify its model by developing the prototype system. Moreover, it has been to verify real-time performance of autonomous agent's architecture for an autonomous mobile robot in the model. The latter research has proved that the implementation of agent architecture level promises to have the potential of finding improved components of autonomous learning and planning allowing efficient parallel processing and quick task switching.

Keywords:

agent architectures, autonomous mobile robots, multi-agent collaboration, real-time performance, rational agency, social agency

Contents

List of Figures

1. Introduction

Due to the spread of computing and networking, the demand of interconnection, and the need to make data accessible at any time and any place are increased. Information environments are composed of distributed, autonomous, heterogeneous components. Recent approach introduces software agents into such environments to deal with these requests. As open environment introduces the dynamism, the complexity and dynamism of the information environments have led to a need for active interfaces and adaptive agents. There are applications such as information access, information filtering, electronic commerce, work-flow management, intelligent manufacturing, education, entertainment, and personal assistant. These applications have a requested common mechanisms of searching, editing, using, presenting, managing, and updating information. The sources of information are autonomous and heterogeneous, and may be added or removed dynamically. The processing mechanisms must be extensible and flexible. Agent technology will perform the above mentioned tasks over unpredictable environments. Further, agents can be constructed locally for each resource and provided to satisfy high-level protocol of interaction.

In this dissertation, we have extended the server-client model, and have proposed a new computational model. In this model, modules and processes of computer systems have a goal to realize autonomy, cooperation, adaptability, and transparency of knowledge to form cooperation among agents in multi-agent system. We research design methodology on four agency models: rational agency, social agency, interactive agency and adaptive agency. And we research mobile agent's function for an efficient computing resource management and an enhancement of services in order to construct a distributed cooperative processing. We propose social agency model which is designed focusing on a role to achieve a common goal, and cooperative method which is knowledge migration with guiding authority between agents in its social agency model based on ontology-based knowledge description following KQML[13]. This proposed model is realized and evaluated by a prototype system of guide activities in a laboratory using a mobile computer and an autonomous mobile robot. And an autonomous agent's architecture including a human interaction for an autonomous mobile robot is developed to compose by independent software components. As a result of executing guide activities in a laboratory, a common goal among agents have been achieved. To enhance this system,an adaptive task scheduling policy and cooperative co-evolving architecture are proposed as future research.

1.1 Motivation

Computational models of intelligence have a number of advantages over problem-solving methods. First, they can be applied when one has only limited knowledge. Second, artificial intelligence can be applied in the context of non-noisy or stationary objective function. So, artificial intelligence works for a well-defined domain if the domain knowledge is limited and defined completely without an interaction from the environment. Distributed artificial intelligence is a subfield of artificial intelligence which has been investigating knowledge models, as well as communication and reasoning techniques that computational agents might need to participate in societies composed of computers and people[43]. The hypothesis underlying this dissertation is that to apply agency model effectively to solve interactive problems with the environment, explicit notions of autonomy must be introduced. Autonomy is defined as autonomous agent described in 2.3.2.

1.2 Current Approaches

It is pointed out that researchers in distributed artificial intelligence are concerned with understanding and modeling action and knowledge in collaborative enterprises[17]. Two main areas of research in distributed artificial intelligence: distributed problem solving and multi-agent systems are distinguished[4]. Distributed artificial intelligence considers how the task of solving a particular problem can be divided among a number of modules that cooperate in sharing knowledge about the problem. All interaction strategies are incorporated as an integral part of the system. Research in multi-agent systems is concerned with the behavior of a collection of autonomous agents aiming at solving a given problem. These problem solvers are autonomous and heterogeneous agents. Jennings[43] proposes a framework that provides a structure for analyzing and classifying most of distributed artificial intelligence research activities. The themes that Jennings will examine are as follows:

- The agent which includes all the elements characterizing an agent involved in a multi-agent system.
- The group perspective to gather the elements that characterize a group of agents which is composed of organization of the group, coordination and planning, and communication and interactions.
- Specific approaches such as open systems, reflection, autonomous agents, and organizational information systems.
- The designer's topics such as methods and implementation techniques for building multi-agent systems, test beds, design tools and applications for distributed artificial intelligence.

1.3 Objectives

The traditional server-client approach to support wireless and mobile applications has resulted in the development of new computational models. Extensions of the server-client model are based on the stationary agents between the mobile client and the fixed server. In a prototype system of guide activities, a new model is required to communicate between the mobile server and the mobile client. The primary goal of this dissertation is to develop a new agency model of cooperation, adaptability, mobility and transparency that combines and extends ideas from rational agent of BDI architecture[45] to improve its ability to communicate in knowledge-level. The following objectives are our milestones in achieving this goal.

- To design and implement a computational model of cooperation, adaptability, mobility and transparency that includes the explicit notion of autonomy.
- To formalize interaction between agents based on coupling design.
- To study the effect of some important characteristics of system design.
- To apply the its model to agent architecture that can be decomposed into layers and tasks in a task-specific domain.

- To evaluate real-time performance of its architecture in a task-specific domain.

- To analyze its efficiency to improve components.

1.4 Methodology

Role theory[3] deals with collaboration and coordination. Roles have also been applied to distributed systems management and to agent and robot systems. Role models are new concepts in object-oriented software engineering that emphasize patterns of interaction. A role defines a position and a set of responsibilities within a role model. A role has collaborators. These are other roles that it interacts with. A role's services and activities are accessible through an external interface. Usually there is a distinct interface for each collaboration path between two interacting roles. In object-oriented software engineering, role models have been developed in response to the following needs:

- Role models emphasize how entities interact with each other.

- Role models describes systems in terms of their patterns of interaction.

- Role models can be instantiated, generalized, specialized, and aggregated into compound models.

- Role models can be dynamic, involving sequencing, evolutions, and role transfer.

In this dissertation, role modeling can be introduced into object-oriented design model.

1.5 Contributions

The main contributions of this dissertation are as follows:

- We have designed and implemented a novel agency model of cooperation, adaptability, mobility and transparency in which the subcomponents of a problem solution are drawn from a collection of agents that collaborate with one another to achieve a common goal.

- This research in the dissertation has proved that the implementation of agent architecture level has the potential of finding improved components of autonomous learning and planning allowing efficient parallel processing and quick task switching.

1.6 Dissertation Outline

The rest of this dissertation is organized as follows. In section 2 we provide background and related work. We describe a survey of agency model and previous related work. In section 3 we provide social agency model for agents collaboration with knowledge migration. We show terminology and goals of this model. This is followed by descriptions of system overview, modeling, evaluations, related work and summary. Section 4 provides autonomous agent's architecture with knowledge migration. We explain design of agent architecture, evaluation, related work and summary. Finally, conclusion in this dissertation is presented in section 5.

2. Background and Related Work

2.1 Agency Models

An agent is a module or a process which has autonomy, cooperation, adaptability, mobility and transparency of knowledge. Agents can work when they are constructed in the development of techniques for designing agents with the above-mentioned features. As a result, agents fulfill their rational, social, adaptive, and interactive functions. In this section, we describe the survey on agency model and the definition of four agency models.

2.1.1 Rational Agency

Rational agency models the abstractions as modal operators in a logic, and describes a set of requirements on its semantics. Rational agency is reviewed in [21][43]. There is theoretical and logical work on this agency. The first attempt to formalize this agency was done by Cohen and Levesque[9]. These works give the grounds for many other formalisms which have been developed[45][52]. Rao and Georgeff[45] present a formalization of the belief, desire, and intention (BDI) architectures for agents. BDI agents must have implicit or explicit representation of the corresponding abstractions or attitudes.

2.1.2 Social Agency

Social agency models societies. Social agency is reviewed in [21][43]. Gasser developed a framework for representing and using organizational knowledge in distributed artificial intelligence. After that, Gasser[17] points out that social concepts of knowledge and action are considered a foundation for distributed artificial intelligence. He offers the duality between agents and the societies in which they exist and function. In Gasser's notion, social agents played the multiple simultaneous roles. Hewitt and Inman[19] develop some enhancements suited to multi-agent systems based on actor concept, which is a means to implement multi-agent systems that supports concurrency and autonomy. The enhancements take the form of organizations of restricted, coordinated sets of actors. Wooldridge and Jennings[59] develop a formalization of Cooperative problem solving is one of the most important capabilities of agent-based systems.

2.1.3 Interactive Agency

Interactive agency models how the agents can commit to one another socially and reason about the social commitments of others. Interactive agency is reviewed in [21][43]. Decker and Lesser[10] propose a generic approach to developing coordination algorithms, which can work in a cooperative environment with preserving execution autonomy. Lux and Steiner[28] study cooperation among agents from an agent's perspective. This generic framework is instantiated to define a variety of cooperation primitives.

2.1.4 Adaptive Agency

Adaptive agency models an adaptive approach to cope with complex environments with agents of limited computational power. Adaptive agency is reviewed in [21][43]. Weiss[58] presents an analysis of reinforcement learning for several agents to coordinate their actions in an environment where they share a common goal but have limited and inexact knowledge. The results can be applied to relatively simple agents. Tan[56] presents on simple hunter-prey experiments with multiple reinforcement learning agents, which share sensory information, policies, and experiences. Sen[47] researches the problem of learning to coordinate and the problem of learning in the absence of explicit sharing of information among the agents.

2.2 Mobile Agents

Mobile agents give a new computational model for mobile computing, which we call the Mobile-Agent Model. This model enables the server-client model to be extended and flexible.

Mobility can be classified into hardware and software mobility[35]. Hardware mobility deals with mobile computing, such as with limitations on the connectivity of mobile computers and mobile IP. A few techniques in mobile computing have security, locating, naming, and communication forwarding. Software mobility can be classified into the mobility of passive data and active data. Passive data represents traditional means of transferring data between computers. It has been employed ever since the first two computers were connected. Active data can be classified into mobile code, process migration and mobile agents. These three classes represent incremental evolution of state transfer. Mobile code, such as Java applets, transfers only code between nodes. Process migration deals primarily with code and data transfer. It also deals with the transfer of authority, for instance access to a shared file system, but in a limited way. Authority is under the control of a single administrative domain. Finally, mobile agents transfer code, data, and especially authority to act within the entire Internet. In our dissertation, we proposed mobile agents which transfer knowledge and authority to guide people between machines.

Mobile agents derive from two fields: agents, as defined in the artificial intelligence community, and distributed systems, including mobile objects and process migration[35]. However, their popularity started with the appearance of the Web and Java. The former opened vast opportunities for applications suited for mobile agents and the latter became a driving programming language for mobile agents. In a Web environment, programming languages focus on platform independence and safety. Innovations in OS services take place at the middle-ware level rather than in kernel. Research in distributed systems has largely refocused from local to wide-area networks. Security is a dominant requirement for applications and systems connected to the Web. In this environment, mobile agents are a very promising mechanism. Typical uses include electronic commerce and support for mobile, distributed computing for which agents overcome limitations posed by short on-line time, reduced bandwidth, and limited storage. Java has proven to be a suitable programming language for mobile agents because it supports mobile code and mobile objects, remote object model and language and run-time safety independent from operating system. Here, IBM's Aglets[26], MEITCA's Concordia[36], Fujitsu's Kafka[41] and Sun's JINI[23] are Java-based agent architectures primarily directed towards building mobile agents which move from machine to machine during execution.

2.3 Related Work

We describe the development based on the above-mentioned agency model.

2.3.1 FRIEND21

FRIEND21 is a national project name of "Future Personalized Information Environment Development", which aimed at an ideal figure of 21th century's human interface in Ministry of International Trade and Industry. It had started in 1988 and had finished in 1994. As a result of this project, human interface architecture, which was called agency model, was developed. This agency model is a executive environment model of meta-ware which is a method of interface design equipped with a real cognitive mechanism so that it may be a dynamic drive by a symbol suitable for a task. It is an integrated operating environment. A memory space of studio is mediated between autonomous modules which are called agents to communicate with each other[16]. In this dissertation, data on shared memory is accessed by independent processes and control functions through socket-base communication in our autonomous agent's architecture of an autonomous mobile robot. If we follow a guideline of FRIEND21, a graphic character and a voice guidance are classified as meta-ware which presents an internal state of data.

2.3.2 Autonomous Agents

Brooks[6] proposed a different approach to creating artificial intelligence that we must incrementally build up the capabilities of intelligent systems. Maes[29] summarizes that the emphasis in autonomous agent architecture is on more direct coupling of perception and action, distribution and decentralization, dynamic interaction with the environment and fundamental mechanisms to cope with resource limitations and incomplete knowledge.

2.3.3 Intelligent and Cooperative Information Systems

The idea of intelligence in information system is important. Intelligent and cooperative information system emerged during the early 1990s. Its system gathers researchers from the fields of information systems, artificial intelligence, data base, software engineering and programming languages. They aim at integrating AI and information system technologies. In [43], Brodie and Ceri indicated that an information system can be considered as intelligent when it incorporates reasoning mechanisms, and that intelligence can be used to improve the quality of user interaction and an information system functionality. The integration of distributed artificial intelligence technology in information systems is appeared.

3. Social Agency Model with Knowledge Migration

A prototype system for guide activities in a laboratory is constructed based on the multi-agent environment. Introducing mobile computing into the multi-agent environment, mobile agents between machines are able to enhance computing efficiency and services. To realize the robust model for required additional services, the following points are considered.

- to execute services on heterogeneous environments
- to develop independent software components

Especially, this section focuses on the basic loop determining both agent's behavior and cooperation among agents. These subsections give some backgrounds by providing terminology and goals, and describe its model, its evaluation and summary.

3.1 Terminology

The terminology of some backgrounds in social agency model is provided:

(1) A **real-world agent** is software agent which has facilities to obtain information from the physical environment or to do something to the environment[40]. This agent can have an autonomous control as an autonomous agent which is defined that an autonomous agent is a system situated within and part of an environment that senses that environment and acts on it so as to effect what it senses in the future[5]. In this paper, a real-world agent is a robotic software agent on an autonomous mobile robot.

(2) **Mobile agents** move dynamically from one machine to another, transferring code, data, and especially authority to act on the owner's behalf within the network[34]. In this paper, a mobile agent with authority can move between the mobile client and the mobile server.

(3) **Knowledge** is state and task of a source agent in Figure 1. It is ontology-based knowledge description to control agents' behavior. And it contributes to the agent's duality described in 2.1.2.

(4) **Knowledge migration** shown in Figure 1 is the act of transferring authority between agents. The transferred authority of the source agent includes the state of a guide and the task to guide a visitor in knowledge-level. After knowledge migration, the destination agent gets the authority and guides a visitor in a laboratory. Throughout this process, agents can share knowledge to achieve a goal of guide activities.

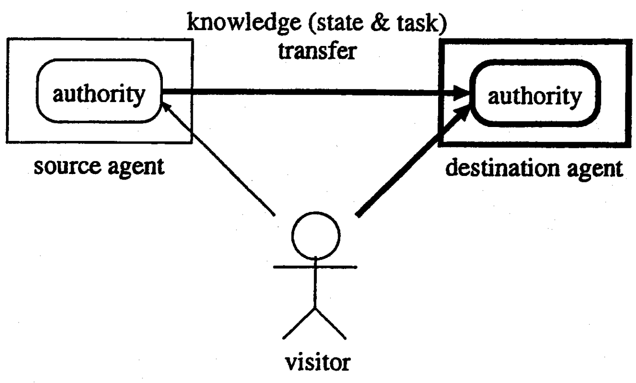

Figure 1. Transferring Authority for Guide Activities by Knowledge Migration

(5) **Autonomy** is defined as autonomous agent described in 2.3.2.

(6) **Social agency** is defined as a model described in 2.1.2.

3.2 Goal

The goals of unifying agent's behavior and cooperation among agents include:

(1) **Cooperation** is a goal of social agency when the multiple processes in the communication between a guide agent and a robotic software agent can work to achieve a common goal that is guiding a visitor to his destination.

(2) **Adaptability** is a goal of social agency when its autonomous software agent can manage both its knowledge and other agents' migrated knowledge to execute its behavior in knowledge-level.

(3) **Mobility** is a goal of social agency when mobile computer and autonomous mobile robot equipped with a network can guide a visitor in a laboratory by knowledge migration between mobile computer and autonomous mobile robot.

(4) **Transparency** is a goal of knowledge migration because the communication and guide activities in a laboratory require to construct transparent knowledge boundaries between real space and virtual space which a computer generates in its display[2]. Real space means the environment in which a visitor, a hand-held mobile computer, and an autonomous mobile robot exist. Virtual space means the graphical map to show that environment in the computer display.

3.3 Overview

Figure 2 illustrates the overview for guide activities based on the multi-agent environment in the experiment rooms of a laboratory. It is composed of an infrared location system, its management server, a hand-held mobile computer and an autonomous mobile robot (Pioneer1 mobile robot) connected by wireless LAN.

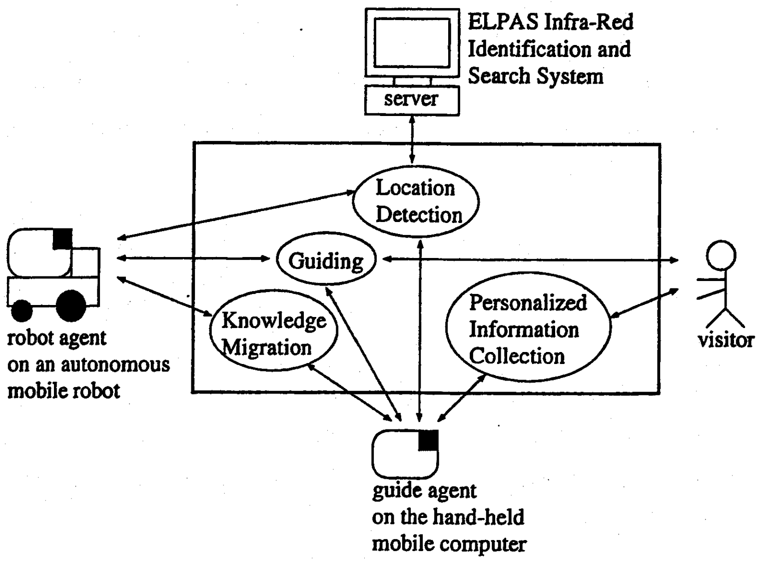

Figure 2. Overview of the Agents' Cooperation Model for Guide Activities in Multi-Agent Environment

3.4 Model

3.4.1 Component

This model's organization is described as follows.

(1) **Guide Agent(GA)**
Figure 3 shows a guide agent that displays a map in a laboratory and a graphic character to guide a visitor on the hand-held mobile computer. This mobile computer is "VAIO PCG-C1" made by Sony. The network connection between this mobile computer and other computers utilizes wireless "WaveLan," which operates at a 1.2 G Hz bandwidth. This system can transmit data at 1 megabit per second.

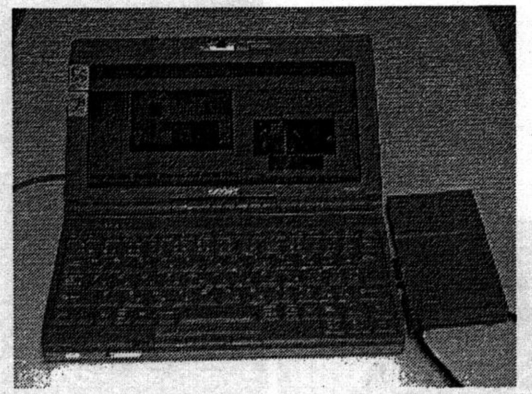

Figure 3. Guide Agent on Hand-Held Mobile Computer

(2) **Robotic Software Agent(RA)**

Figure 4 shows a robotic software agent that assists visitors to utilizes an autonomous mobile robot made by ActivMedia. This autonomous mobile robot shown in Figure 4 has seven sonars, an encoder, an electrical compass, and two motors. It is controlled by an operating system only used for this robot (PSOS), and this OS is installed in a control board on the robot. A client system terminal is connected to the OS. This terminal is a notebook type personal computer, "SOLO" made by Gateway, in which Red Hat Linux release 5.1 is installed. This client system receives a packet from PSOS including input from seven sonars, an encoder and a compass data. The data transmission rate is one packet per 100 m sec. This client system was developed by using Saphira Libraries to connect with PSOS.

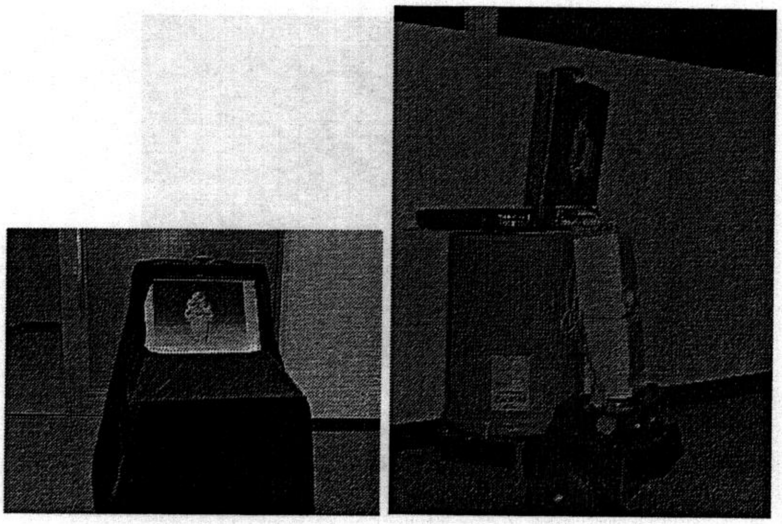

Figure 4. Prototype Autonomous Mobile Robot for Guide Activities

(3) **Location System(LS)**

It is utilized to detect the location of the robotic software agent and the visitor's mobile computer. The location system can read the infrared emission from a badge put on a mobile robot and a mobile computer. The infrared location system's readers on the ceilings of the hallways detect the mobile computer's and mobile robot's location. This location information is updated on the location system's server.

3.4.2 Interaction

In the multi-agent environment, there are four kinds of interaction as follows.

(1) **Personalized Information Collection**

A visitor inputs his research interest on the hand-held mobile computer according to a guide agent's instruction.

(2) **Knowledge Migration**

Authority for guiding a visitor is transferred by knowledge migration between a guide agent and a robotic software agent.

(3) **Guiding**

A visitor is navigated by a guide agent and a robotic software agent to a visitor's destination in a laboratory.

(4) Location Detection
ELPAS infra-red identification and search system detects the physical locations of the mobile computer and the mobile robot.

3.4.3 Knowledge Migration

(1) Design
Figure 5 shows ontology-based knowledge sharing. Each agent has its own knowledge constrained by its role and its obligation. The location system detects a guide agent's and a robotic software agent's locations, and notifies each location to each agent. A guide agent collects visitor's personalized information which is context for guiding. This knowledge of a guide agent is transferred to a robotic software agent. After transferring, related knowledge to migrated one is combined and the robotic software agent's knowledge is newly formed. After that, the robotic software agent can decide its behavior in the environment and have authority to guide a visitor.

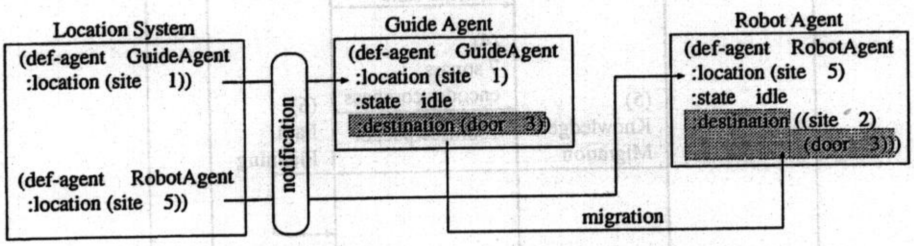

Figure 5. Ontology-Based Knowledge Sharing and Knowledge Migration

(2) Protocol
Using knowledge migration, authority of guiding is transferred from a guide agent to a robotic software agent. Figure 6 indicates a protocol and processes to execute authority. This protocol trace is described as follows.

1. to detect and avoid obstacles by using input from the seven sonars at one packet per 100 m sec.

2. to transmit a site number sent by the location system through the communication layer to the behavior layer.
3. to transmit an internal state from the behavior layer to the communication layer for updating this state.
4. to transmit input from seven sonars, a compass value and an encoder value from the action layer to the behavior layer at one packet per 100 m sec.
5. to migrate knowledge of a guide agent from the communication layer to the behavior layer.
6. to execute path planning in the behavior layer, and to transmit motor control from the behavior layer to the action layer.

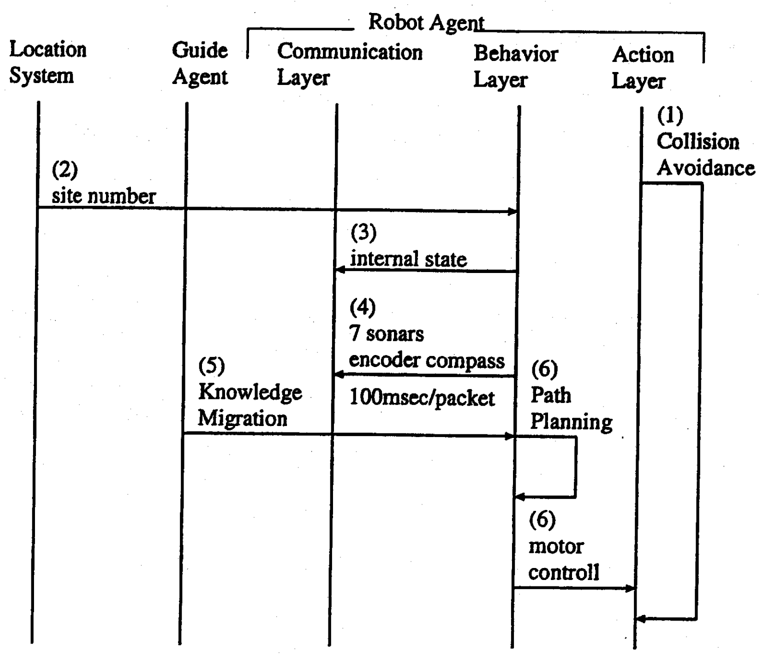

Figure 6. Protocol Trace to Execute Authority which is Migrated from Guide Agent

3.4.4 Formalization

A key objective of this work is to develop a formalism that provides useful and powerful abstractions for the agent designer, while having a well-defined interpretation in terms of real systems. In this section, we focus on the relationship between agents. A formal representation of the agents' cooperation model[42] is introduced.

$$AgentSystem =< Agents, Environment, Coupling >$$

where:

(1)

$$Agent =< State, Input, Output, Process >$$

- State is the set of properties (values, true propositions) that completely describes the agent.
- Input and output are subsets of state whose variables are coupled to the environment.
- Process is an autonomously executing process that changes the agent's state.

(2)

$$Environment =< State, Process >$$

- The environment has its own process that can change its state, independent of the actions of its embedded agents.

(3) Coupling is a mapping of an agent's input and output from/to the environment's state.

Based on this formalization[42], It shows precisely how agents' interaction and communication can be proven to guarantee the navigation for a visitor in a laboratory. The formal model is defined by the following five elements.

Definition 1.
The multi-agent model, M, is a structure:

$$M = (A, E, C)$$

where:

- A = $\{GA, RA\}$ is a set of agents.

- E = $\{LS\}$ is a set of environment which has the its own process that can change its state, independent of the actions of A.

- C is coupling model that the agents has a mapping of input and output to or from the environment.

Definition 2.
Coupling model, C, is a structure:

$$C = (I, O)$$

where:

- I = $\{INTEREST\}$ is a set of inputs from the environment.

- O = $\{DESTINATION\}$ is a set of outputs to the environment.

Definition 3.
GA model, A(GA), is a structure:

$$A(GA) = (S, I, O, P)$$

where:

- S = $\{$*attention, calculating, pushing, waiting*$\}$ is a set of GA's states[11].

- I = $\{INTEREST,\ SITE\}$ is a set of inputs to the environment.

- O = $\{KNOWLEDGE\}$ is a set of outputs to the environment.

- P = $\{$*instruction, guiding, migration*$\}$ is a set of processes that change the agent's states98.

Definition 4.
RA model, A(RA), is a structure:

$$A(RA) = (S, I, O, P)$$

where:

- S = {*idle, transmission, guiding, goal*} is a set of RA's states.
- I = {*GA(KNOWLEDGE), SITE*} is a set of inputs to the environment.
- O = {*DESTINATION*} is a set of outputs to the environment.
- P = {*wandering, guiding*} is a set of processes that change the agent's states.

Definition 5.
LS model, E(*LS*), is a structure:

$$E(LS) = (S, P)$$

where:

- S = {*GA(SITE), RA(SITE)*} is a set of LS's states.
- P = {*detection, notification*} is a set of processes that change LS's states.

3.5 Evaluation

Social agency model is evaluated by guiding a visitor in a prototype system. Knowledge migration between a guide agent and a robotic software agent can guarantee the agent's duality described in 2.1.2. As for its duality including people, they conducted experiments, and proved a visitor's guiding context persistence in a laboratory in [44].

3.6 Related Work

We survey and review [18][21][54] concerning multi-agent systems. Socket-based communications with text messages such as FIPA[14] or KQML[13] like our research are employed. They allow large communities of agents to be built. In these systems, images and complex data structures would need to be explicitly serialized by each agent. IBM's Aglets[26], MEITCA's Concordia[36], Fujitsu's Kafka[41] and Sun's JINI[23] are Java-based agent architectures primarily directed towards building mobile agents which move from machine to machine during execution. In this dissertation, knowledge, which is described based on ontology, is migrated between agents, and this migration transfers authority of guiding people between machines.

3.6.1 The Open Agent Architecture

The Open Agent Architecture[30] makes it possible for software services to be provided through the cooperative efforts of distributed collections of autonomous agents. Communication and cooperation between agents are brokered by one or more facilitators, which are responsible for matching requests, from users and agents, with descriptions of the capabilities of other agents. In this architecture, a blackboard style of communication is introduced as a global data repository which can be used cooperatively by a group of agents. Cooperation among the agents of the Open Agent Architecture system is achieved through messages expressed in a common language, ICL. ICL is the interface, communication, and task coordination language shared by all agents, regardless of what platform they run on or what computer language they are programmed in. In this dissertation, ontology-based knowledge description is used to communicate with agents through socket-base knowledge migration. Our social agency model in guide activities is realized without a blackboard style of communication. Our autonomous agent architecture of an autonomous mobile robot in social agency model, which is described in Section 4, uses a blackboard style of communication between multiple processes.

3.7 Summary

In summary, this section provided our social agency model's design and formalization on agents cooperation with knowledge migration which was introduced into a prototype system of guide activities in a laboratory. Our goal was cooperation to achieve a common goal that is guiding a visitor to his destination, adaptability to manage both its knowledge and other agents' migrated knowledge to execute its behavior in knowledge-level, mobility to guide a visitor in a laboratory by knowledge migration between mobile computer and autonomous mobile robot, and transparency to construct transparent knowledge boundaries between real space and virtual space which computer generates in its display. This was accomplished using psychological experiments using a robot interface model with a lifelike agent in order to construct a relationship between humans and robots[44].

In subsection 3.4 we described an agent system model which is focused on agents, environment and coupling that is a mapping of an agent's input and output from/to the environment's state. We demonstrated the model as the knowledge level of cooperative interaction between four agents within machines.

In the final subsection, related work with cooperation and multi-agent system is investigated. Socket-base communication with text message and mobile agent which is composed of data and code with authority are employed to construct multi-agent system. We emphasize that our social agency model is specialized towards the needs of executing services on heterogeneous environments and developing independent software components.

4. Autonomous Agent's Architecture

This section will describe a design and its evaluation of real-time control architecture for an autonomous agent in guide activities. An autonomous agent is an autonomous mobile robot which is a robotic software entity. An autonomous agent decides its behavior from environment inputs, controls interface agent with warning voice and a life-like graphic character, and communicates with other agents in knowledge-level connected by TCP. This agent is introduced into a prototype system for guide activities in a laboratory to evaluate its real-time performance. To realize the robust model for required additional services, the following points are considered.

- to develop independent software components
- to negotiate agent's behavior between software components using protocols

4.1 Agent Architecture

This section describes the implementation of determining both agent's behavior and cooperation among agents. This software architecture shown in Figure 7 has three layers of "communication layer," "behavior layer" and "action layer." The functions in each layer are described as follow.

4.1.1 The control layers

(1) Communication Layer
This layer has two functions. One is to manage four internal states for a robotic software agent's behavior: "Idle," "Transmission," "Guiding" and "Goal" according to the interruption of its state change. Its interruption details describe in a next subsection. Another is to control interface agent which displays life-like graphical character and outputs a voice guidance according to these internal states.

(2) Behavior Layer
This layer has two functions. One is to manage several inputs which are a site number given by the location system, data from the seven sonars, a

compass value, an encoder value, an internal state, and a door number as a destination. Another is to execute path planning to direct the real-world mobile agent's behavior.

(3) Action Layer
This is composed of Saphira control architecture, three input components from and one output component to Saphira control architecture, and the management system for input information from the location system by TCP.

Figure 7 illustrates the structure of a robotic software agent's software architecture within Pioneerl mobile robot holding the Saphira's control architecture[24]. The robotic software agent's software architecture organizes multiple processes to the cooperative architecture. Running processes in each layer communicate with each other using UNIX sockets. These modules of encoder, sonars, compass inputs, and motors' outputs using Saphira Libraries are controlled in task switching manner using finite-state machine. Interface agent control of graphic character's behavior and voice guidance are implemented on Java applets in Web. Internal state data on shared memory is accessed by control functions(encoder, sonars, and compass), path planning, and internal state management. The robotic software agent receives its environment inputs through packet communication protocol, migrated knowledge, and a site number sent by the location system through TCP. It manages its present location and running direction, executes path planning, and carries out run operation to find a destination.

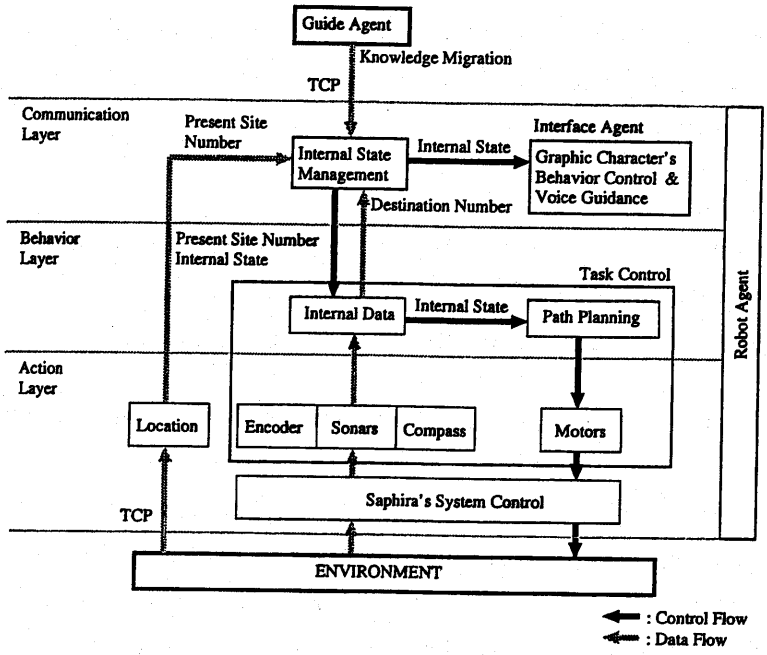

Figure 7. Agent's Software Architecture on Autonomous Mobile Robot

4.1.2 The flow of control

There are three generic control paths that are illustrated in Figure 8 a-c. The bottom-up control mechanism can be always worked in the action layer. Figure 8 a) describes the reactive path. Emergency situations or situations that can be dealt with by a routine without requiring explicit planning are recognized and handled at the action layer. When the agent is running and recognizes the change of site, control takes the local planning path. Situations handled by using this path are requiring local path planning. A plan is generated and executed by activating sequences of a procedure(Figure 8 b)). The cooperative path additionally involves the communication layer, as they require coordination with other agents in knowledge level.

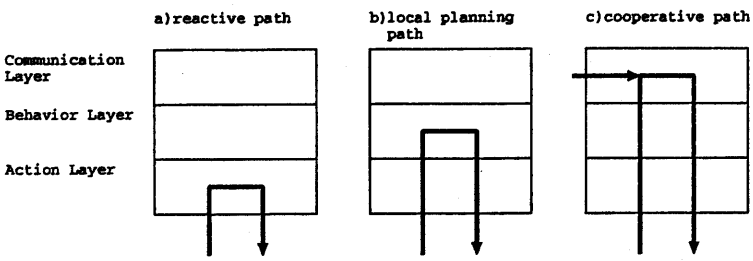

Figure 8. The Flow of Control

4.2 Saphira's System Control

Saphira's system control is a set of routines that interpret sensor readings relative to a geometric world model, and a set of action routines that map robot states to control actions. Registration routines link the robot's local sensor readings to its map of the world, and the Procedural Reasoning System sequences actions to achieve specific goals. Pioneerl mobile robot provides the Saphira control architectures built on top of a synchronous, interrupt-driven OS[24]. Micro-tasks are finite-state machines that are registered with the OS. The OS cycle per 100 ms through all registered finite-state machines performs one step in each of them. Because these steps are performed at fixed time intervals, all the finite-state machines operate synchronously. In Figure 8, task control's component is designed and implemented based the above finite-state machines. Figure 9 indicates the design of task division.

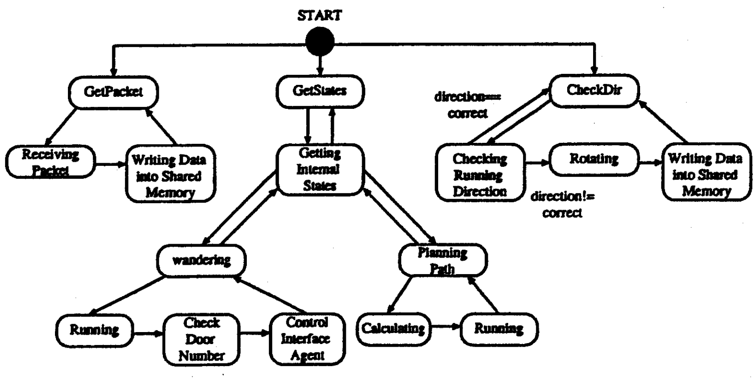

Figure 9. Design of Task Control to Determine Basic Loop of Robotic Software Agent

Following this design, micro-task is described in Figure 10. *sfInitProcess* is a top level of task control. Task executes its operation sequentially by changing state.

```
void main(int argc, char **argv){
    sfOnConnectFn(myConnectFn);   /* register a connection function*/
    sfOnStartFn(myStartupFn);     /* register a startup function*/
            |
            |
            |
}

void myConnectFn(void){
    sfInitProcess(GetPacket,"GetPacket");
    sfInitProcess(GetStates,"GetStates");
    sfInitProcess(CkeckDir,"CheckDir");
}

void GetPacket(void){
    switch(process_state){
      case sfINIT;
      case 20: PacketReceive();
               process_state=30;
               break;
      case 30: WriteDataIntoSharedMemory();
               process_state=20;
               break;
    }
}
```

Figure 10. Description of Micro-Task in Task Control

4.3 Protocol

4.3.1 Internal State Transition

Figure 11 illustrates an internal state transition model. This model is described as the following cycle.

1. In the idle state shown in Figure 12, the robotic software agent goes back and forth in a corridor, stops in front of each door and outputs a voice guidance.

2. In the transmission state shown in Figure 13, the robotic software agent meets visitors in the idle state. And the guide agent on the visitor's mobile

computer migrates authority to the robotic software agent after sending a transmission command from the guide agent to the robotic software agent. In the case of no visitor with a mobile computer or no need for migration, the robotic software agent continues in an idle state.

3. In the guiding state shown in Figure 14, the robotic software agent starts to guide the visitor to the destination.

4. In the goal state, the robotic software agent reaches its destination and sends a transmission end command to a guide agent. After that, the robotic software agent is in the idle state.

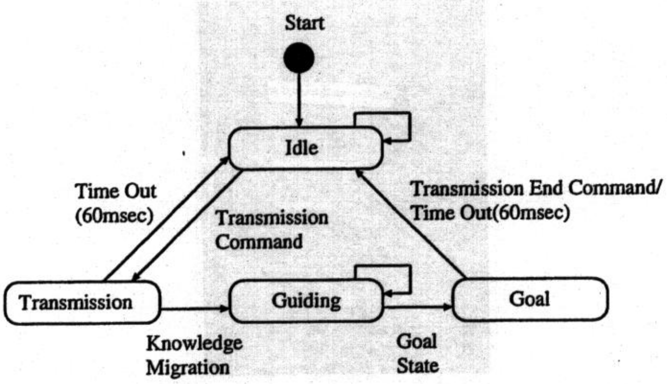

Figure 11. State Transition to Determine Basic Loop of Agent's Behavior

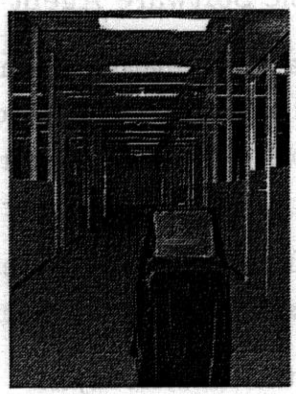

Figure 12. Idle

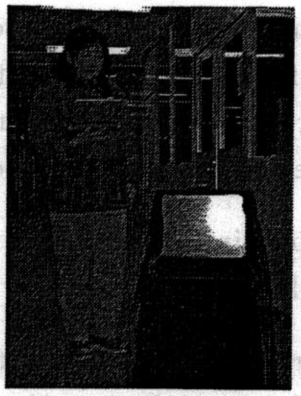

Figure 13. Transmission

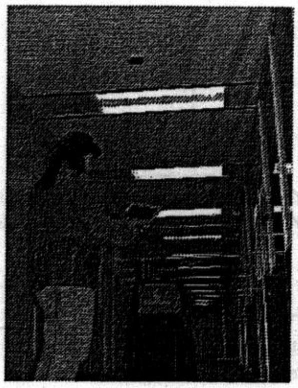

Figure 14. Guiding

4.3.2 Transition of Robotic Software Agent's Behavior State

Following this internal state transition, path planning is executed in the behavior layer, and a forward, backward or rotation control command is sent to the action layer to control the two motors. Figure 15 illustrates the trigger protocol to transit an internal state in the communication layer as follows.

1. to send an internal state from the behavior layer to the communication layer for updating the robotic software agent's internal state.

2. to be in the idle state in the behavior layer.

3. to send a transmission command from a guide agent through the communication layer to the behavior layer.

4. to be in the transmission state in the behavior layer. If the agent cannot change its state to the idle state within 60 m sec, its state will default to the idle state (back to above 2.).

5. to transfer knowledge for guiding from a guide agent through the communication layer to the behavior layer.

6. to be in the guiding state in the behavior layer.

7. to execute path planning, and to run forward a destination location.

8. to send a goal state through the communication layer from the behavior layer to a mobile computer.

9. to send a transmission end command from a guide agent through the communication layer to the behavior layer. If the agent cannot receive this command within 60 m sec, it will default to the Idle state in the behavior layer (back to above 2.).

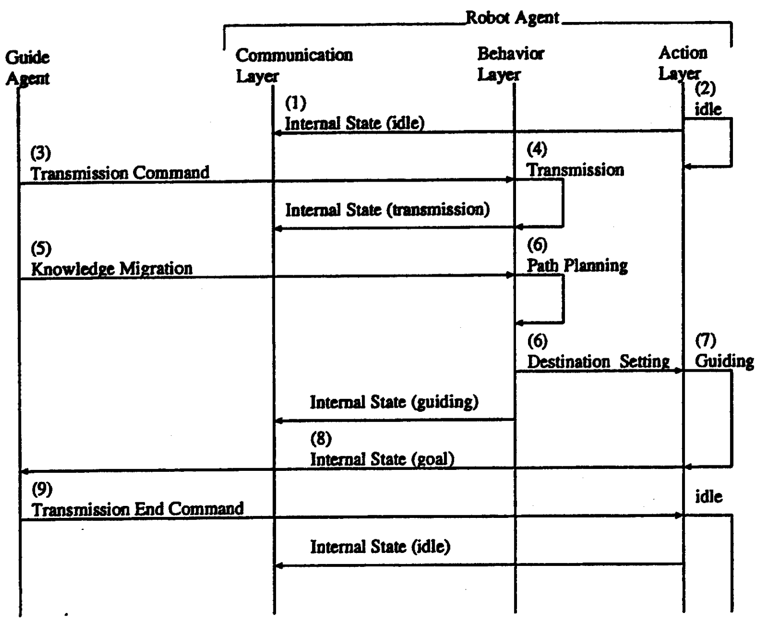

Figure 15. Protocol Trace to Transit Agent's Behavior State

4.4 Path Planning

Figure 16 and Figure 17 indicate the division of a mobile robot's running space in the east and west directions. The robotic software agent goes back and forth between a starting point of a site number "2" and an ending point of a site number "7" which exist rooms and exhibitions available for guiding. A visitor can walk between a site number "1" and "8" with the hand-held mobile computer. In the guiding state, an agent calculates the distance from the present location to a destination's location by executing path planning in the behavior layer and orders a running command for this distance to the action layer. In this path planning, one path is defined as the section between site numbers indicating changed points. The agent runs until it receives a site number of a destination by checking an order of receiving a site number, and it runs from the entrance of a final site and the destination location by checking its encoder value. The robotic software agent runs until a site number includes a destination, and when it enters a destination site area, it runs the distance between the entrance of a site point and the destination by using the encoder. The following procedure is taken.

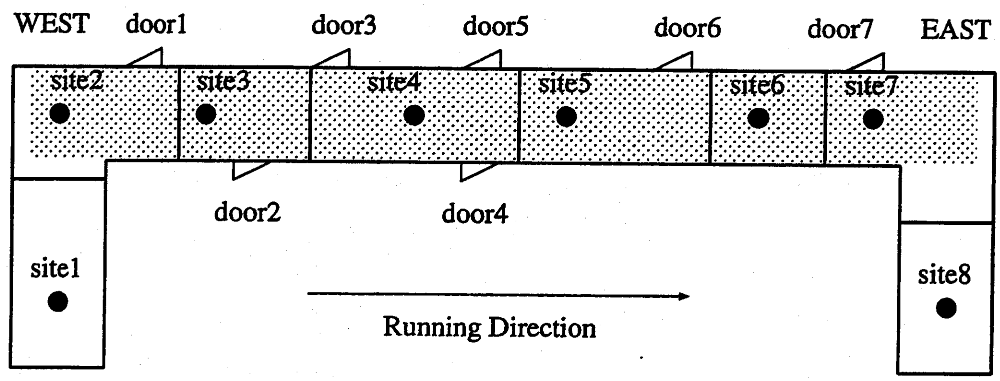

Figure 16. Space Division in East Direction where Robotic Software Agent is Running

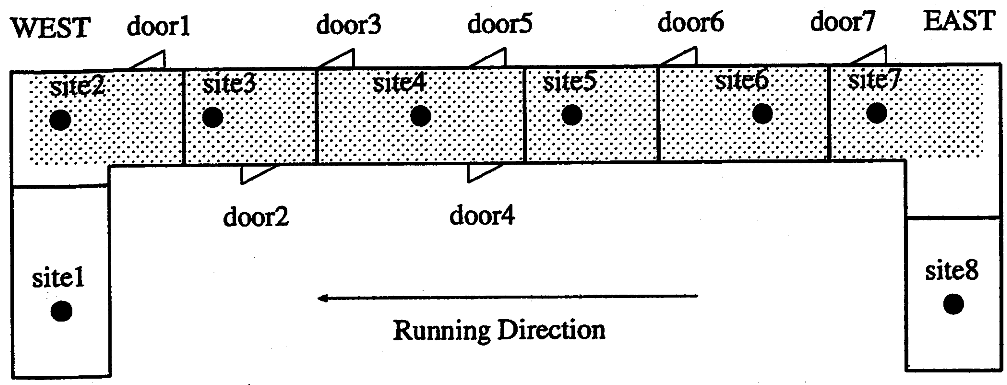

Figure 17. Space Division in West Direction where Robotic Software Agent is Running

1. checking a running direction
   If a running direction is different from that of the destination from the present location, the robotic software agent rotates 180 degrees.

2. running until a site number includes a destination

3. calculating the distance from the entrance of a site including the destination to the destination. Goal-Distance means this distance. For example, when a running direction is east and the destination is door4, the entrance of a site indicates the first point to detect site4 closely connected with site3.
   When Goal-Site is detected,

$$GoalDistance \leftarrow (D1, ..., Dn)$$

D: the distance between an entrance point in a site including the destination and the destination
(UNIT:mm)

n: a door number including the destination

Goal-Site: a site number including the destination

4. arranging running direction by using the location system
   In updating a site number, a heading degree is calculated by a theta value

and a compass value in order to minimize the gap of the encoder value with respect to straight running.

4.5 Evaluation

4.5.1 Description of the experiments

Navigation experiments were conducted on a robotic software agent with a multi-agent system under 0 percent of TCP packet loss. The interruption of task switching and receiving data through TCP, and the inputs from the environment affect a robotic software agent's control in the architecture. The robot runs in a corridor between sites number "2" and "7." The robot's running is compared between a program for straight forward running and a program for a robotic software agent in axis(x,y) and a theta value. X and Y are the coordinates of the robot in a 2-D Cartesian coordinate system and theta is its orientation. Y indicates a value of a straight running distance. X indicates a value of sideways, which is a gap from a straight running. All x,y coordinates are in millimeters. Theta indicates a gap of a running direction between the starting point and the present point. Theta is in degrees from -180 to 180. The following experiment involves running in the east and west directions. In the graphs plotted for these experiments, "STANDARD" means a result of a program for straight forward running and "EVALUATION" means a result of a program for a robotic software agent's execution.

4.5.2 Results

Results of the experiments are described as follows.

(1) East Direction
Figure 18 indicates location X-Y. The horizontal value means Y. The vertical value means X. Figure 19 indicates location theta-Y. The horizontal value means Y. The vertical value means theta.

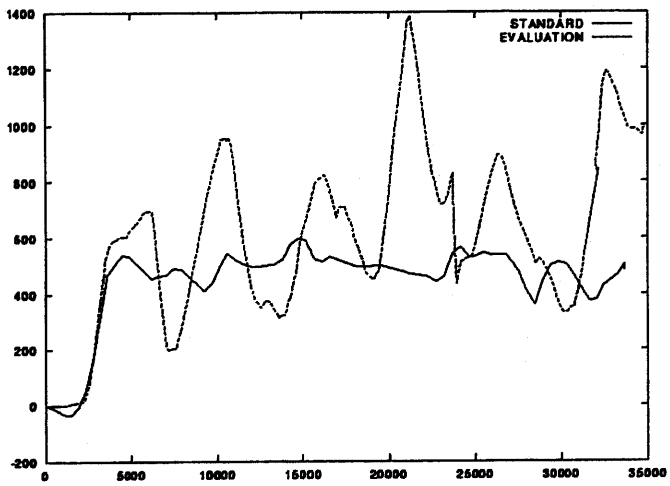

Figure 18. AXIS X and Y in East Direction of Robotic Software Agent

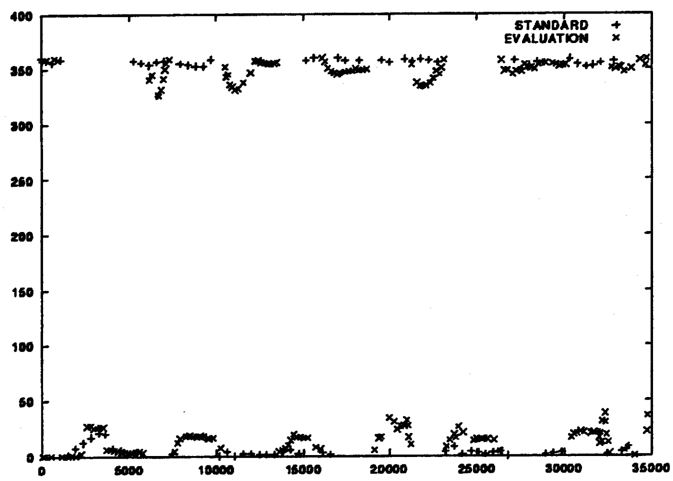

Figure 19. Theta Value in East Direction

(2) West Direction
Figure 20 indicates location X-Y. The horizontal value means Y. The vertical value means X. Figure 21 indicates location theta-Y. The horizontal

value means Y. The vertical value means theta.

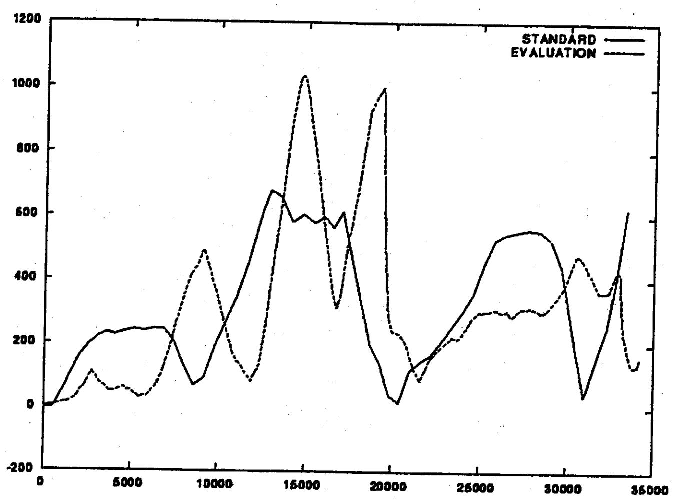

Figure 20. AXIS X and Y in West Direction of Robotic Software Agent

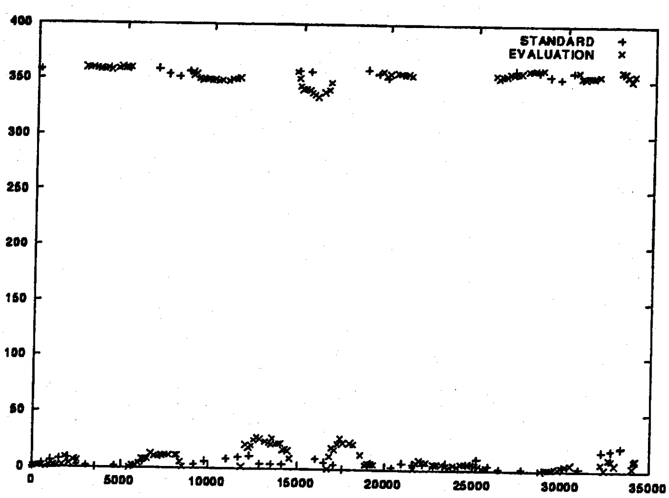

Figure 21. Theta Value in West Direction

Due to the gap in degrees between standard and evaluation, the robotic software agent in the physical world can run against a wall even if collision avoidance management is executed. This gap is attributed to running at un-uniform velocity and slipping because task switching and receiving data through TCP interrupt its behavior control.

4.6 Related Work

4.6.1 BDI architecture

Rao and Georgeff[45] considered a particular agent framework. These individual rational agents incorporate certain "mental attitudes" of Belief, Desire and Intention (BDI). These are used to represent, respectively, the information, motivational and deliberative states of the agent, and determine the system's behavior. The core components of the BDI framework are rational agents[22]. These are autonomous entities, which execute independently, and have complete control over their own internal behavior. The core elements of BDI agents are as follows. Beliefs correspond to the information that the agent has understood the world. The set of beliefs is typically incomplete and may be incorrect with respect to the true situation. Desires are the agent's high-level goals. They need not to be achievable simultaneously, but are usually consistent. Intentions essentially represent a subset of the agent's desires that it has committed to achieve. A practical development approach still does not consider the maintenance of formal properties. So far, the need for high-level logic-based languages capturing the key components of the BDI model remains.

4.6.2 INTERRAP architecture

Based on BDI architecture by Rao and Georgeff, the INTERRAP architecture has been developed in the MAS Group of DFKI[37]. INTERRAP is a layered architecture. This agent consists of three components which are a World Interface (WIF), a Knowledge Base (KB), and a Control Unit (CU). WIF provides the agent's sensors, communication, and actors liked to its environment. KB and the control unit has three layers which are the Behavior-Based Layer (BBL), the Local Planning Layer (LPL), and the Cooperative Planning Layer (CPL). BBL has a role of fast reaction on crucial events. All actions which a agent can perform are encoded in the BBL. The LPL plans its behavior and decided sequences of behavior patterns to achieve his goal. When some goals cannot be achieved without the cooperation with other agents, an agent communicates and builds cooperation plans. In our dissertation, knowledge migration can execute cooperation between agents.

4.7 Summary

In summary, this section provided an autonomous agent's architecture with knowledge migration in social agency aimed at communicating with other agents in knowledge-level. This architecture was designed by two criteria;(1)to develop independent software components and (2)to negotiate agent's behavior between software components using protocols, and evaluated in it real-time performance.

In subsection 4.1, 4.2, 4.3 and 4.4, we described a design of an autonomous agent architecture which is focused on above mentioned two criteria. We demonstrated that a task of controlling inputs and output, a task of managing path plan, and a task of presenting a guidance can work cooperatively in its architecture and that knowledge migration can contribute to the communication between agents in order to achieve a common goal.

In subsection 4.5 real-time performance of our autonomous agent architecture was investigated. We conducted experiments that the robot's running is compared between a program for straight forward running and a robotic software agent in axis(x,y) and a theta value.

In the final subsection, related work in a layered approach is investigated. BDI architecture and INTERRAP architecture have no socket-base communication with text message and mobile agent which are employed to construct cooperative multi-agent system. We emphasize that our approach of knowledge migration can overcome the limitation of adaptability in controlling over agent's internal behavior.

5. Conclusions

5.1 Summary

This dissertation has addressed a domain limitation of artificial intelligence that reduces its effectiveness when applied to increasingly complex problems, namely they lack the explicit notion of autonomy. Our goal has been to find computational extensions of rational agent to improve its ability to communicate in knowledge-level. The primary issue has been how to develop a new agency model of cooperation, adaptability, mobility and transparency.

To accomplish this mission, we designed, formalized and analyzed a computational model of cooperation, adaptability, mobility and transparency to achieve a common goal. Section 3 provided our social agency model's design and formalization on agents cooperation with knowledge migration which was introduced into a prototype system of guide activities in a laboratory. Our goal was cooperation to achieve a common goal that is guiding a visitor to his destination, adaptability to manage both its knowledge and other agents' migrated knowledge to execute its behavior in knowledge-level, mobility to guide a visitor in a laboratory by knowledge migration between mobile computer and autonomous mobile robot, and transparency to construct transparent knowledge boundaries between real space and virtual space which computer generates in its display. This was accomplished using psychological experiments using a robot interface model with a lifelike agent in order to construct a relationship between humans and robots. We described an agent system model which is focused on agents, environment and coupling that is a mapping of an agent's input and output from/to the environment's state. We demonstrated that as the knowledge level of cooperative interaction between four agents within machines. The usability of our social agency model was investigated. We conducted experiments for the inspection that as a result of the migration the robot can inherit the context from the interactions between a user and an agent so that a relationship between a user and a robot is formed. Related work with cooperation and multi-agent system is investigated. Socket-base communication with text message and mobile agent which is composed of data and code with authority are employed to construct multi-agent system. We emphasize that our social agency model is specialized towards the needs of executing services on heterogeneous environments and developing independent software components.

Section 4 provided an autonomous agent's architecture with knowledge migration in social agency aimed at communicating with other agents in knowledge-level. This architecture was designed by two criteria;(1)to develop independent software components and (2)to negotiate agent's behavior between software components using protocols, and evaluated in it real-time performance. We described a design of an autonomous agent architecture which is focused on above mentioned two criteria. We demonstrated that a task of controlling inputs and output, a task of managing path plan, and a task of presenting a guidance can work cooperatively in its architecture and that knowledge migration can contribute to the communication between agents in order to achieve a common goal. Real-time performance of our autonomous agent architecture was investigated. We conducted experiments that the robot's running is compared between a program for straight forward running and a robotic software agent in axis(x,y) and a theta value. This result have showed that this implementation of agent architecture level promises to have the potential of finding improved components of autonomous learning and planning allowing efficient parallel processing and quick task switching. Related work in a layered approach is investigated. BDI architecture and INTERRAP architecture have no socket-base communication with text message and mobile agent which are employed to construct cooperative multi-agent system. We emphasize that our approach of knowledge migration can overcome the limitation of adaptability in controlling over agent's internal behavior.

Finally, this research in the dissertation has proved that the implementation of agent architecture level promises to have the potential of finding improved components of autonomous learning and planning allowing efficient parallel processing and quick task switching.

5.2 Future Research

Throughout this dissertation we have suggested a number of possible directions for future research into the design of computational models of agency. To conclude, we now briefly expand on a number of these ideas.

5.2.1 Adaptive task scheduling policy

According to a priority of task or semantics of knowledge for behavior, it is possible to control task switching by the policy of task allocation. Task is classified into a task independent from a context of agent's behavior and a continual task dependent on a context of agent's behavior. To maintain a continual procedure of task switching, queuing theory[57] is applied to design a scheduling policy of task allocation. Under a condition that a requested transaction rate α (request per second) is greater than a queuing rate β (input per second), a response time at average T is calculated using Kleinrock's proof as follows:

$$T = \frac{1}{\alpha - \beta} \tag{1}$$

When a number of a task which is possible to be allocated is n, each task has queuing system, but it cannot be worked parallel, the above-mentioned formula(1) can produce a response time at average as follows:

$$T/n = \frac{1}{n\ \alpha - n\ \beta} \tag{2}$$

Executing time of a continual task dependent on a context of agent's behavior in queuing at time t is calculated by the following algorithm:

n : a number of task allocation
t : a time to start calculating a number of task allocation
continual_task : a continual task dependent on a context of agent's behavior
ExecutingTime : a continual task executing time(unit:second)
Job : the total time of continual task executing
Buffer : queing buffer

```
procedure JOBTIME;
begin
  i = 0;
  while termination condition not satisfied do
    begin
      select cotinual_task from Buffer(i);
      add ExecutingTime to Job_{t-1};
      i = i + 1;
      evaluate termination in Buffer(i);
    end
end{JOBTIME};
```

Task allocation is executed at a present time t based on the above-mentioned formula(2) as follows:

$$T/n = \frac{1}{n\ \alpha\ - n\ \beta} < Job_{t-1} \qquad (3)$$

According to the above-mentioned procedure and formula(3), a number of task allocation is decided and task switching will be done.

5.2.2 Cooperative co-evolving architecture

By knowledge migration, agents' knowledge is shared, reused and coupled. Co-evolutionary model is introduced into learning of behaviors for autonomous mobile robot in knowledge-level. It is possible to generate new knowledge for complex behavior in order to adapt the un-descriptive environment. We propose a co-evolving model with knowledge sharing and utilization. Based on two layered approach of learning classifier system, a cooperative co-evolutionary architecture which will realize its model is constructed(Figure 22).

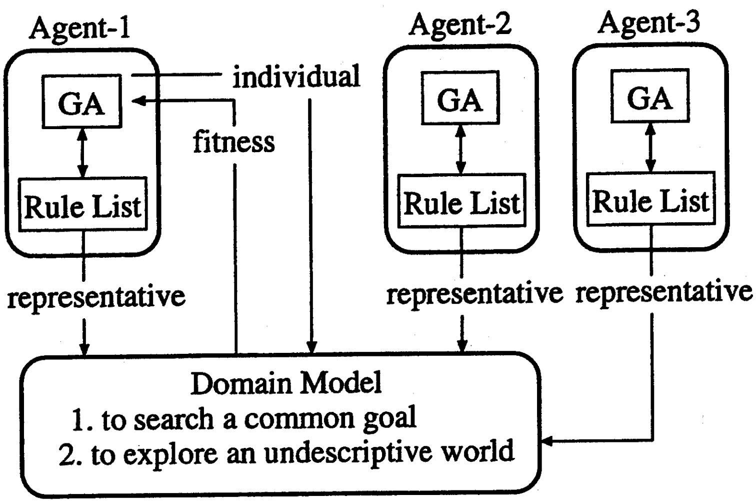

Figure 22. Cooperative Co-evolutionary Architecture

Each agent is constructed based on a layered learning classifier system. It has its own knowledge. When each agent selects a best rule, each agent offers its best rule into shared space and each agent get a best rule from shared knowledge. This procedure is as follows:

FOR Each agent s **THEN BEGIN**
Choose representatives from agents;
$i \leftarrow$ individual from agent s requiring evaluation;
Evaluate fitness of collaboration by applying i to target domain;
Assign fitness of collaboration to agent s;
END

The above-mentioned architecture will be introduced into knowledge management of an agent.

In these future research, we will investigate its efficiency of modeling and design in agency model.

Acknowledgements

Especially, the author thanks the comments on my doctor's thesis from Dr. Masatsugu Kidode, Dr. Hideaki Takeda, Dr. Shunsuke Uemura and Dr. Katsumasa Watanabe to finish my five years' academic research in Graduate School of Information Science, Nara Institute of Science and Technology(NAIST). The author thanks WWW to search, know and gather the information on computer science, information science and world-wide agent technology. The author acknowledges lectures, discussions and supports from Professors, Associate Professors, Instructors and Secretary in NAIST. The author gratefully acknowledges IEEE and ACM because my papers were accepted and appealed in world-wide academic communities.

References

[1] Hideki Asoh, Satoru Hayamizu, Isao Hara, Yoichi Motomura, Shotaro Akaho and Toshihiro Matsui. Socially Embedded Learning of the Office-Conversant Mobile Robot Jijo-2. Fifteenth International Joint Conference on Artificial Intelligence. Proceedings Volume 2. August. 880–885. 1997.

[2] Steve Benford, Chris Greenhalgh, Gail Reynard, Cris Brown and Boriana Koleva. Understanding and Constructing Shared Spaces with Mixed-Reality Boundaries. ACM Transactions on Computer-Human Interaction. volume 5. number 3. September. 185–223. 1998.

[3] B. J. Biddle and E. J. Thomas. Role Theory: Concepts and Research. New York. R. E. Krieger Publishing Co. 1996.

[4] A. H. Bond and L. Gasser. Readings in Distributed Artificial Intelligence. Morgan Kaufmann. San Mateo, CA. 1988.

[5] J. Bradshaw, editor. Software Agents. MIT Press. 1997.

[6] R.A. Brooks. Intelligence without representation. Artificial Intelligence 47. 139–159. 1991.

[7] R.A. Brooks. A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation. Vol. 2. No. 1. 14–23. 1996.

[8] W. Burgard, A.B. Cremers, D. Fox, D. Haehnel, G. Lakemeyer, D. Schulz, W. Steiner and S. Thrun. The Interactive Museum Tour-Guide Robot. Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence. Outstanding Paper Award. 1998.

[9] Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. Artificial Intelligence 42. 213–261. 1990.

[10] Keith S. Decker and Victor R. Lesser. Designing a family of coordination algorithms. In Proceedings of the International Conference on Multi-agent Systems. 73–80. 1995.

[11] S. Fels, S. Sumi, T. Etani, N. Simonet, K. Kobayashi and K. Mase. Progress of C-Map: A Context-Aware Mobile Assistant. Proceedings of the AAAI Spring Symposium on Intelligent Environments. March. 1998.

[12] Maier Fenster, Sarit Kraus and Jeffrey Rosenschein. Coordination without communication: Experimental validation of focal point techniques. In Proceedings of the International Conference on Multi-agent Systems. 102–108. 1995.

[13] T. Finin, Y. Labrou and J. Mayfield. KQML as an agent communication language. In J. Bradshaw, editor. Software Agents. MIT Press. 1997. http://www.cs.umbc.edu/agents/introduction/kqmlacl.ps.

[14] FIPA: Foundation for Intelligent Physical Agents. FIPA home page. 1999. http://www.fipa.org/.

[15] Karen Frenkel. An interview with Robin Milner. Communications of the ACM 36(1). 90–97. 1993.

[16] FRIEND21: Future Personalized Information Environment Development. Human interface Architecture Rules Document. PIE, MITI. 1994.

[17] L. Gasser. Social conceptions of knowledge and action: DAI foundations and open systems semantics. Artificial Intelligence 47. 107–138. 1991.

[18] L. Gasser. An overview of DAI. Distributed Artificial Intelligence: Theory and Praxis. Kluwer Academic. 1992.

[19] Carl Hewitt and Jeff Inman. DAI betwixt and between: From "intelligence agents" to open systems science. IEEE Transactions on Systems, Man, and Cybernetics 21(6). 1409–1419. 1991.

[20] John H. Holland. ADAPTATION IN NATURAL AND ARTIFICIAL SYSTEMS. The MIT Press. 1993.

[21] M. Huhns and M. Singh. Readings in Agents. Morgan Kaufmann. 1997.

[22] N. R. Jennings and M. Wooldridge. Applying Agent Technology. Artificial Intelligence. 47. 1991.

[23] Jini architecture overview. Technical report. Sun microsystems. July. 1998. http://java.sun.com/products/jini/whitepapers/architectureoverview.pdf.

[24] Kurt G. Konolige. Saphira Software Manual. Version 6.1e. April. 1998.

[25] Erno Kovacs, Klaus Rohrle, and Bjorn Schiemann. Adaptive Mobile Access to Context-aware Services. Proceedings of First International Symposium on Agent System and Applications, and Third International Symposium on Mobile Agents. IEEE Computer Society. 190–201. 1999.

[26] D. Lange and M. Oshima. Programming and Deploying Java Mobile Agents with Aglets. Computer and Engineering Publishing Group. 1998. http://www.trl.ibm.co.jp/aglets/.

[27] Michael Littman, Anthony Cassandra and Leslie Kaelbling. Learning policies for partially observable environments: Scaling up. In Proceedings of the 12th International Conference on Machine Learning. 362–370. 1995.

[28] Andreas Lux and Donald Steiner. Understanding cooperation: An agent's perspective. In Proceedings of the International Conference on Multi-agent Systems. 261–268. 1995.

[29] P. Maes. Designing Autonomous Agents. MIT Press. Cambridge. MA/Elsevier. Amsterdam. 1990.

[30] David L. Martin, Adam J. Cheyer, and Douglas B. Moran. The Open Agent Architecture: A Framework for Building Distributed Software Systems. Applied Artificial Intelligence. vol. 13. 91–128. 1999.

[31] John McCarthy. Ascribing mental qualities to machines. In Martin Ringle, editor. Philosophical Perspectives in Artificial Intelligence. Harvester Press. Brighton, England. 1979.

[32] John McCarthy. Elephant2000: A programming language based on speech acts. Computer Science Department. Stanford University. http://www-formal.stanford.edu/jmc/elephant.html. 1992.

[33] Robin Milner. Elements of interaction. Communications of the ACM 36(1). 78–89. 1993.

[34] Dejan S. Milojicic, William LaForge and Deepika Chauhan, *Mobile Objects and Agents(MOA)* , Proceedings of the Fourth USENIX Conference on Object-Oriented Technologies and Systems(COOTS98), Santa Fe, New Mexico, (April), 27-30, 1998.

[35] Dejan S. Milojicic, Fred Douglis, Yves Paindaveine, Richard Wheeler and Songnian Zhou. Process Migration. HP Labs 1999 Technical Report. HPL-1999-21. 1999. http://www.hpl.hp.com/techreports/1999/.

[36] Mitsubishi Electric Information Technology Center America. White Paper: Mobile agent computing. Technical report. MEITCA. 1998. http://www.meitca.com/HSL/Projects/Concordia/.

[37] Jorg P. Muller. The Design of Intelligent Agents. Lecture Notes in Artificial Intelligence 1177. Lecture Notes in Computer Science. Springer. 1996.

[38] Jorg P. Muller. Control Architectures for Autonomous and Interacting Agents: A Survey. Lecture Notes in Artificial Intelligence 1209. Lecture Notes in Computer Science. Springer. 1997.

[39] David J. Musliner, Edmund H. Durfee, and Kang G. Shin. World modeling for the dynamic construction of real-time control plans. Artificial Intelligence 74. 83–127. 1995.

[40] K. Nagao and J. Rekimoto. Agent augmented reality: A software agent meets the real world. Proceedings of the Second International Conference on Systems (ICMAS-96). AAAI Press. 228–235. 1996.

[41] T. Nishigaya. Design of multi-agent programming libraries for Java. Technical report. Fujitsu Laboratories, Ltd.. 1997. http://www.fujitsu.co.jp/hypertext/free/kafka/paper/.

[42] Jim Odell and William Tozier. Agents and Complex Systems. ASA/MA Tutorial. The Joint Symposium ASA/MA99, First International Symposium on

Agent Systems and Applications (ASA'99), Third International Symposium on Mobile Agents (MA'99). October 3-6. 1999.

[43] G.M.P.O'hare and N.R.Jennings. Foundations of Distributed Artificial Intelligence. Sixth-Generation Computer Technology Series. JOHN WILEY SONS INC.. NSBN 0-471-00675-0. 1996.

[44] T. Ono, M. Imai, T. Etani and R. Nakatsu. Construction of Relationship between Humans and Robots. Transactions of Information Processing Society of Japan. Vol. 41. Number 1. January. 158–166. 2000.

[45] A. S. Rao and M. P. Georgeff. Modeling Agents within a BDI-Architecture. In International Conference on Principles of Knowledge Representation and Reasoning (KR). Cambridge. Massachusetts. April. Morgan Kaufmann. 473–484. 1991.

[46] Jeffrey S. Rosenschein and Gilad Zlotkin. Designing conventions for automated negotiation. AI Magazine. 29–46. 1994.

[47] Sandip Sen, Mahendra Sekaran and John Hale. Learning to coordinate without sharing information. In Proceedings of the National Conference on Artificial Intelligence. 426–431. 1994.

[48] Yoav Shoham. Agent-oriented programming. Artificial Intelligence 60(1). 51–92. 1993.

[49] Jaime Simao Sichman, Rosaria Conte, Yves Demazeau and Cristiano Castelfranchi. A social reasoning mechanism based on dependence networks. In Proceedings of the 11th European Conference on Artificial Intelligence. 188–192. 1994.

[50] Munindar P. Singh. A critical examination of the Cohen-Levesque theory of intentions. In Proceedings of the 10th European Conference on Artificial Intelligence. 364–368. 1992.

[51] Munindar P. Singh. A semantics for speech acts. Annals of Mathematics and Artificial Intelligence 8(I-II). 47–71. 1993.

[52] Munindar P. Singh. Multiagent Systems: A Theoretical Framework for Intentions, Know-How, and Communications. Springer-Verlag. Heidelberg, Germany. 1994.

[53] L. Steels. Exploiting analogical representation. In Designing Autonomous Agents. MIT Press/Elsevier. Cambridge, MA. 71–88. 1990.

[54] K. Sycara. Multiagent systems. AAAI Magazine. 19(2). 1998.

[55] Milind Tambe, Lewis Johnson and Wei-Min Shen. Adaptive agent tracking in real-world multi-agent domains: A preliminary report. In Proceedings of the AAAI Spring Symposium on Adaptation, Coevolution, and Learning in Multiagent Systems. AAAI Press. Menlo Park, CA. 1996.

[56] M. Tan. Multi-agent reinforcement learning: Independent vs. cooperative learning. In Proceedings of the 10th International Conference on Machine Learning. 330–337. 1993.

[57] Andrew S. Tanenbaum. Modern Operating Systems. Prentice-Hall, Inc. 1992.

[58] Gerhard Weiss. Learning to coordinate actions in multi-agent systems. In Proceedings of the International market model for distributed configuration design. AI EDAM 9. 125–133. 1995.

[59] Michael Wooldridge and Nick Jennings. Formalizing the cooperative problem solving process. In Proceedings of the 13th International Workshop on Distributed Artificial Intelligence. 403–417. 1994.

List of Publications

A. Journal Papers

[1] Noriko Etani, Hiroshi Kudoh, Keita Satoh, Masaya Itoh and Terunobu Fujino, "Applying Human-Behavior Analysis and Design of Man-Machine Interaction, -A Proposal of Human-Centered System Design-," *Progress in Human Interface*, Vol.3 No.1, The Society of Instrument and Control Engineers, pp.23-29, 1994(in Japanse).

B. International Conference papers

[1] Noriko Etani, "Adaptive Agency Model for A Real-World Mobile Agent," *Digest of 3rd IEEE Workshop on MOBILE COMPUTING SYSTEMS and APPLICATIONS(WMCSA'2000)*, Sponsored by the IEEE Computer Society, Technical Committee on the Internet(TCI) and Technical Committee Operating Systems(TCOS), In cooperation with ACM SIGMOBILE and the USENIX Association, December 7-8, 2000.

[2] Noriko Etani, "Robot Media Communication: A Real-world Guide Agent to Construct Transparent Knowledge Boundaries Between Real and Virtual Spaces," *Intelligent Agent Technology: Systems, Methodologies and Tools (Proceedings of First Asia-Pacific Conference on Intelligent Agent Technology)*, WORLD SCIENTIFIC PUBLISHING COMPANY PTE LTD., ISBN981-02-4054-6, pp.53-57, December 14-17, 1999.

[3] Noriko Etani, "Robot Media Communication: An Interactive Real-World Guide Agent," *Proceedings of First International Symposium on Agent Systems and Applications(ASA '99), Third International Symposium on Mobile Agents(MA '99)*, the IEEE Computer Society, ISBN 0-7695-0340-3, pp.234-241, October 3-6, 1999.

C. Other Conference papers

[1] Noriko Etani, "Real-Time Performance of Autonomous Agent's Architecture With Shared Migrating Knowledge," *Proceedings of Computer System Symposium*, IPSJ Symposium Series Vol.2000, No.13, Information Processing Society of Japan, ISSN 1344-0640, pp.131-137, November 15-November 16, 2000.

[2] Noriko Etani, "Robot Media Communication: A Real-world Multi-Agent System for Guide Activities to Construct Transparent Knowledge Environment," *Eighth Workshop on Multi-Agent and Cooperative Computation*, Japan Society for Software Science and Technology, November 30-December 1, 1999(in Japanese).

[3] Noriko Etani, "An Experiment on Enhancement of Human Elements which Affect Software Reliability : An Observation on Participatory Design," *Proceedings of the tenth Symposium on Human Interface*, The Society of Instrument and Control Engineers, pp.135-142, 1994(in Japanese).

[4] Noriko Etani, "VEIL: The Utilization of Meta Information to support the Collaborative Framework," *Interactive System and Software I*, Japan Society for Software Science and Technology, Kindai-Kagaku Sha, ISBN4-7649-0226-5, pp.81-88,1994(in Japanese).

D. Technical Reports

[1] Noriko Etani, "Conference Report on ASA/MA99," *Journal of JSAI*, Vol.15 No.2 2000/3, pp.380, Japanese Society for Artificial Intelligence, March 1, 2000(in Japanese).

[2] Noriko Etani, "Publicity of Engineering," *Interactive Essay*, *Journal of IPSJ*, Vol.41 No.1(2000.1), Information Processing Society of Japan, http://www.ipsj.or.jp/magazine/interessay.html, February 3, 2000(in Japanese).

[3] Noriko Etani, "Conference Report on ASA/MA99," *Journal of IPSJ*, Vol.41 No.1, Information Processing Society of Japan, pp.107, January 15,

2000(in Japanese).

[4] Noriko Etani, Tameyuki Etani and Kenji Mase, "Design and Implementation of Robot-Media to Realize Seamless Information Environment," *IPSJ SIG Notes(Distributed Processing System)*, Vol.99 No.30, ISSN 0919-6072, Information Processing Society of Japan, pp.37-42, 1999(in Japanese).

[5] Noriko Etani, Hideaki Takeda and Toyoaki Nishida, "Using Classifier System to Learn Adaptive Strategies for Collision Avoidance," *IPSJ SIG Notes(Intelligence and Complex System)*, 111-3, Information Processing Society of Japan, pp.15-20, 1998(in Japanese).

[6] Noriko Etani, Naoko Tosa and Ryouhei Nakatsu, "A Script Management for Interactive Story," *1997 Spring Annual Conference*, A16-10, The Institute of Electronics, Information and Communication Engineers, pp.382, 1997(in Japanese).

[7] Noriko Etani, "Object-Oriented Analysis and Design Methdology, and Its Direction," *Advanced Technology and Its Application for Software Productivity Progress, Eighth Symposium on Software Productivity Progress*, LECTURE, The Japan Institute of System Research, 1994(in Japanese).

[8] Miwako Doi, Noriko Etani, Michitaka Hirose, Hiroo Iwata, Yu Shibuya, Atsuya Yoshida and Hiroshi Tamura, "Project Proposal and Discussion on Developer Project of Human Interface which you bet," *Report on Human Interface Symposium 1993*, Human Interface News and Report, Vol.9, The Society of Instrument and Control Engineers, pp.99-110, pp.211-216, 1994(in Japanese).