Human-Robot Interface Architecture for Distributed Environments

Yasushi Nakauchi  Kenji Kawasugi  Toyoshi Okada
Nobuyuki Yamasaki  Yuichiro Anzai

Department of Computer Science, Keio University
3-14-1, Hiyoshi, Kohoku-ku, Yokohama 223 Japan
E-mail: {nakauchi, kawasugi, okadaa, yamasaki, anzai}@aa.cs.keio.ac.jp

Abstract—In order to support cooperative work of human, the research of computer-supported cooperative work has been evolved. And many groupware have been developed in recent years. In the cooperative work of human, the task may involve in the transportation of physical entities that conventional groupware can not handle. We propose an interface architecture for the transportation of physical entities in distributed environments by utilizing mobile robot.

In this paper, we propose the multi-agent model that can represent human-robot interaction in distributed environments by extending the multi-agent model of groupware toolkit Michele we have developed. And we have implemented the interface system based on the extended multi-agent model by utilizing autonomous mobile robot Einstein developed in our laboratory. In the paper we describe the multi-agent model, agent description language, implementation issues and application for the office automation.

1. Introduction

In general the work that carried out in general office or an organization is cooperative work. In recent years, the research of CSCW (Computer Supported Cooperative Work) has been evolved and many groupware (computer system which support cooperative work of group) have been developed [1]. For the communication, groupware usually provides the interface that utilizes text, graphics, video, audio, etc. But in general, when humans are doing cooperative work, information passed among them are not only electronic information, but also physical information. For instance, when you are designing a merchandise, color, texture and weight are important factors and you can not eliminate these characteristics in discussing the design. And these physical information are very difficult to describe if only by using electronic information. Therefore, we think the actual physical entity should be transferred from one to the others as physical information. Then we can discuss with better information. By supporting transportation of physical entities, much wider range of cooperative work can be supported.

In order to transport physical information, we can use mobile robots. And transportation of information can be seen as the cooperative work of human and mobile robots. Human may tell a task (i.e. take and carry an object to a certain person) to mobile robots and robots may also ask a task (i.e. pick up an object from the robot and process it) to humans. We will show an example of human-robot cooperative work in distributed office environment in figure 1.

There are all sorts of communication between human and robot. The characteristics of human-robot communication can be classified according to whether communication is based on synchronous interactive communication or it's based on asynchronous communication. It can also be classified according to whether communication is via electronic information or via physical information. We describe each of four communication types briefly below.

Synchronous/electronic communication is real-time communication via computer network or via radio. Example systems include tele-operation with virtual reality technology. Asynchronous/electronic communication is communication such as tele-operation of a planetary rover that an operator expects a message issued will be received by the robot with delay of time. Synchronous/physical communication is real-time interactive communication. That is passing from human’s hand to robot’s hand — cash dispensers, vending machines, time recorders, etc. Asynchronous/physical communication is asynchronous interaction. That is to say, raw material is supplied to a robot where it will be processed and produce merchandise with time. The procedures in office automation application can be classified as shown in figure 1. And a series of human-robot cooperative work consists of these four types of communication.

The interface systems for the robot have been studied, which include the interface for teaching procedures to robots [2], the interface for tele-operation [3] and the tele-operating interface for controlling semi-autonomous
1. Worker A calls a mobile robot somewhere in the office building. (Async/Elec)
2. The robot comes to worker A’s office.
3. Worker A hands documents to the robot. (Sync/Phys)
4. Worker A commands the robot to carry verbal message along with the documents. (Sync/Elec)
5. The robot moves to worker B’s office.
6. The robot hands the documents to worker B. (Sync/Phys)
7. The robot gives messages from worker A to worker B. (Sync/Elec)

Fig. 1. Procedures of the office automation application.

robot [4]. But these systems never consider the four types of human-robot communication we have mentioned above.

In this paper we propose RT-Michele the interface system for supporting human-robot cooperation in distributed environments. We have developed a groupware toolkit called Michele (Multi-agent Interface with Communication by HecTic ELEments) [5], which supports human cooperative work based on asynchronous communication. Human cooperative work can be represented rigorously with the multi-agent model of Michele. RT-Michele is an expansion of Michele. RT-Michele provides not only asynchronous communication, but also real-time interactive communication. We have implemented a prototype of RT-Michele. We also applied it to office automation application that is cooperation of human and mobile robot Einstein we have developed.

II. DESIGN

A. Multi-Agent Model of RT-Michele

Within the cooperative work of human and robots, the number of users and robots is unlimited and they execute concurrently in a distributed environment. To represent concurrent behavior, we can use multi-agent model such as Actor[6] and ABCL/1[7]. But in these models, the method of communication between agents is too primitive to represent the heterogeneous human-robot communication we have described in section I.

We have developed groupware toolkit called Michele[5] for supporting cooperation of humans only via asynchronous communication. Michele provides multi-agent model that can represent asynchronous communication between users. We will propose the multi-agent model for human-robot cooperation by extending the multi-agent model of Michele.

For every user and robot to work simultaneously, each agent has its own processing power and local persistent memory called fields. The contents of an agent's fields define its internal state. Each agent also has procedures called methods that specify its behavior. Any communication from one agent to another is realized as a methodcall defined by a method.

As shown in Fig. 2, an agent can be a system-agent, a user-agent, or a robot-agent. A system-agent is an entity realized as a process of a computer system. A system-agent can represent electronic information such as specification of a task, plan structure of a robot, and an office document. A user-agent is the combination of a user and a MI (Message Interpreter). A robot-agent is the combination of any kind of autonomous robot and TI (Task Interpreter). Corresponding to functions of the robot, a robot-agent has a set of methods. Only through methodcalls, other agents can invoke and ask services for the robot.

To deal with both asynchronous and synchronous real-time communication, each user has his or her own user-environment and each robot has its own robot-environment. An agent can create a meeting-environment dynamically (See Fig. 2). An environment is a notion to distinguish asynchronous communication and synchronous communication. Communication between two environments is asynchronous, while communication within one environment is synchronous.

A user-environment is a metaphor of the private office of a user. If a user is in his or her own user-environment, he or she can execute a task represented as an agent within the user-environment through synchronous interactive communication. A robot-environment is a metaphor of working environment for an autonomous mobile robot. The information of the task that the robot is currently executing, and the information of allotted and queued tasks to the robot are represented as system-agents in the robot environment.

A meeting-environment is a metaphor of a council room for humans or a shared space for human and robots. This notion is incorporated to realize synchronous interactive communication among human and robots. Every time a human or a robot decides to communicate with companions (human or robots) through interactive communication, he or she creates a meeting-environment, then migrates into the meeting-environment, and ask the companions to join the meeting. When the companions migrate to the meeting-environment, they can communicate through interactive communication. In the case of interactive communication between a human and a robot, they may talk to each other by using voice recognition and voice synthesis systems, and they may give or take physical entities between them. The state transition chart of
user/robot-agent are shown in Fig. 3.

There are some characteristics for physical information that discriminate it from electronic information. Physical information is unique in the world and can not be duplicated. It represents the information that describes the object itself, such as size, weight, color, etc., and also the information that describes the task attached to the object such as whom it is carried to and how it is processed.

Therefore, to handle physical information, we define physical-agent as a special system-agent. The physical-agent manages characteristics of the object. A robot receives an object from a human, then a physical-agent is instantiated by the robot corresponding to each object. When the object is passed to a human, the physical-agent corresponding to the object can be deleted. The specification of physical-agents can be described by the programmer using the multi-agent description language MDL/C++.

B. Multi-Agent Description Language: MDL/C++

We have developed multi-agent description language called MDL/C++ to describe the multi-agent model of RT-Michele. MDL/C++ enables the user to customize his user-environment and construct various kinds of applications. A program of MDL/C++ consists of agent declarations. An agent declaration is composed of declarations of fields and methods as shown in Fig. 4. The fields specify the internal state of the agent. And the methods can be called by other agents.

The syntax of MDL/C++ is based on C++ [8] and the procedure of methods can be written in C++. The declaration of physical-agent is declared by using declarator named p_agent instead of declarator named agent.

An agent is instantiated by operator new. And the pointer to the created agent will be returned as the return value. The method whose name is same as that of agent is called constructor as same as C++, and the constructor is executed as soon as the agent is instantiated. Only these agents (called primary agents) that have a constructor can be instantiated by a user directory. Only primary agents can be registered as the applications of RT-Michele and they start the application.

The communication between agents in the same environment is described as a method call as same as C++. On the other hand, the communication between agents residing in the different environment is described by special function called migrate as shown in Fig. 5. When an agent executes function migrate, the agent will be mi-
agent AGENT_NAME {
    public:
        field char fieldname1[] = "initial_value";
        field int fieldname2  = initial_value;
        field agent AGENT_NAME1 *agent_pointer;

    method METHOD_NAME() {
        // Procedures of the method
    }
}

if(migrate(environment_name) == NULL){
    // Procedures executed after migration
    TI->enqueue(Message);
} else {
    // Procedures executed if migration fails
}

Fig. 3. The state transition chart for establishing meeting-environment.

Fig. 4. Declaration of an agent in MDL/C++.

grated to the environment specified in the argument\textsuperscript{1}. After migration the function \textit{migrate} returns NULL as the return value.

As described in section II, user-agent is the combination of a user and a MI. Thus, the methodcall between system-agent and user-agent will be executed as follows. From the viewpoint of system-agent, it will be the methodcall to MI. And from the viewpoint of user, it will be the selection of the menu shown by MI respectively. There are two kinds of interaction between system-agent and user-agent, namely the presentation and inquiry of information. In order to support these interaction, MI has two kinds of methods \textit{show} and \textit{ask}. Method \textit{show} presents the information to a user and method \textit{ask} inquires the information to a user through text editor, bitmap editor or menu selection.

III. IMPLEMENTATION

A. Mobile Robot Einstein

We have implemented the prototype of interface system by using mobile robot \textit{Einstein} developed in our laboratory (See Fig. 6). \textit{Einstein} has one CPU board, two pulse
motors, four supersonic sensors to measure the distance of four directions, one wireless modem to communicate with workstations, and a manipulator to hold objects. \textit{Einstein} is equipped with multi-thread real-time operating system called \textit{PULSER} [9].

B. Consideration of Distributed Environments

At the implementation level, key issues are how to maintain the internal state of an agent and how to realize communication between agents in distributed environments.

The internal state of an agent should be kept persistent all through the execution of cooperative work. Therefore, we have implemented the internal state of an agent as an e-mail in each user's mail folder. The e-mail has fields and their values as the internal state of the agent.

There are two kinds of communication between agents. One is the communication between agents in the same environment. The other is the communication between agents in different environments. And further more, the communication of agents in the same environment can be classified, such as if it is between user-agent and system-agent, between user-agent and robot-agent, or between system-agent and robot-agent.

The communication between user-agent and system-agent in the same environment is implemented as the procedure call of the appropriate method. The procedures of methods written in MDL/C++ are compiled to the procedures of C++ by the preprocessor. And then they are compiled to executable object by C++ compiler. Therefore, a methodcall is implemented as a methodcall of C++.

The communication between user-agent and robot-agent and also between system-agent and robot-agent are implemented as the methodcall to TI (Task Interpreter). When TI receives a message from agents, TI sends the message to \textit{Einstein} via radio.

On the other hand, communication between agents in different environments is performed by the migration of agent over different environments. The migration of agents is implemented as transportation of e-mails via computer network. While a daemon called ICS (Inter-user-environment Communication Support system) resides in each user/robot-environment. And when ICS receives incoming e-mail and if it contains the internal state of an agent, then ICS reactivates the e-mail as the agent.

IV. Application

We have implemented the office automation application described in section I. The application of RT-Michele should be modeled by the multi-agent model and described by MDL/C++ then compiled. We will describe how the application is implemented in the following.

We have designed the system-agent named \textit{orderly} to support a series of the cooperative work by a user and \textit{Einstein}. When a user (hereafter worker \textit{A}) invokes the office automation application, \textit{orderly}-agent is instantiated in the user-environment. Once \textit{orderly}-agent has instantiated, it receives worker \textit{A}'s order and looks for the appropriate mobile robot. Then the agent migrates to the robot-environment and tells worker \textit{A}'s order to the robot via TI.

When \textit{Einstein} comes to worker \textit{A}'s office, \textit{Einstein} creates a meeting-environment and enters it, then asks worker \textit{A} to enter the meeting-environment. Worker \textit{A} will realize it by robot icon as shown in Fig. 7. There may be more than one robots in the office building. And several robots may come to worker \textit{A}'s office and may ask to enter their meeting-environments at the same time. Worker \textit{A} can choose one of the robots by selecting a menu as shown in Fig. 7.

After worker \textit{A} and \textit{Einstein} enters the same meeting-environment, they can have a synchronous communication. In this application, worker \textit{A} gives documents to \textit{Einstein}, then \textit{Einstein} instantiates the physical-agent called \textit{document}-agent. After \textit{document}-agent has instantiated, \textit{document}-agent automatically asks worker \textit{A} for the information necessary to handle the documents as shown in Fig. 7. Then \textit{Einstein} moves to worker \textit{B}'s office with the documents. And similar procedures will repeat at worker \textit{B}'s office.

V. Conclusion

In this paper we have proposed the multi-agent interface architecture for human-robot cooperation in distributed environments. Cooperative works of human and robots
can be modeled by the multi-agent model of RT-Michele and can be described by MDL/C++. With this framework, a series of cooperative work can be controlled by agents. Furthermore, it is possible that an intelligent agent executes a task in the cooperative work on behalf of a user.

With the multi-agent model of RT-Michele, we can represent asynchronous/synchronous communication and also communication via electronic/physical information. We believe that the model of RT-Michele is one of the basic frameworks for modeling the interaction between human and robots. We will extend and continue our study to seek better way of communication for human-robot cooperation.

REFERENCES


