Human Media Interface System for the Next Generation Plant Operation

Riichiro Mizoguchi[*], Akio Gofuku[**] , Yoshitsugu Matsuura[***]
Yoshihiko Sakashita[****] , Mikio Tokunaga[*****]
miz@ei.sanken.osaka-u.ac.jp
[*] The Institute of Scientific and Industrial Research, Osaka University
[**] Department of Systems Eng., Okayama University
[***] Mechatronics Development Center, Ishikawajima-Harima Heavy Industries Co., Ltd.
[****] Information Technology R&D Center, Mitsubishi Electric Corporation
[*****] Engineering Department, Nippon Mitsubishi Oil Corporation

ABSTRACT

This is an overview paper of a project on "Development of a human interface for the next generation plant operation" running as a subproject of the MITI-funded Human Media project. The goals of the project include develop the next generation human-centered interface for plant operation. We adopted an oil-refinery plant as a task domain and have been developed an interface system consisting of Interface Agent which is responsible for monitoring plant and for direct interaction with operators, Virtual Plant Agent responsible for 3D virtual display of the plant, Semantic Information Presentation Agent for presentation semantically interpreted information based on sophisticated model of the plant, Ontology Server for standardizing of Agents' understanding of the plant and Distributed Collaboration Infrastructure for enabling agents to perform collaboration. Design philosophies and a project overview is presented.

1. INTRODUCTION

Human Media project is a MITI(Ministry of International Trade and Industries)-funded national project started in January, 1997. This is an overview paper of a project on "Development of a human interface for the next generation plant operation" running as a subproject of the Human Media project. It is MITI's intention that research on concrete topics to realize the idea of Human Media technology should be conducted to make the results visible and easy-understandable rather than basic research about Human Media in general. Two years and a half have passed since it got started and we have a lot of fruitful results which we would like to share with people out side of the project.

The philosophy underlying the project is summarized as the coined term "Human Media" which stands for the advanced friendly media technology for humans using three representative media such as Knowledge media, Virtual media and "Kansei" media. "Kansei" is a Japanese term whose rough meaning is the sixth or seventh sense for satisfaction, comfort, beauty, softness, etc. Technologically, the heart of Human Media is an integration of the above three kinds of advanced information technology to come up with an innovative human-centered interface.

We see four major trends of paradigm shifts in computer technology:

(1) From Computer-centered to Human-centered
(2) From Processing-centered to Information-centered
(3) From Form-oriented to Content-oriented
(4) From Centralized control to Distributed control

The first two are based on deep reflection on the long history of computer-centered and processing-centered research which have never been good in human-computer interaction aspects, since such technology forces a human to approach machines/systems or allow, at best, addition of an ad-hoc interface on top of each system. We need to change paradigmatically to come up with an innovative and essentially better human interface technology. The concepts of Human-centered and Information-centered technologies are key concepts of such an enterprise. The true man-machine systems which are what we need in the coming information age do require an open architecture involving humans who need computers to help them facilitate their daily activities. The heart of Human Media project is compliant with these paradigm shifts, more strongly, it is an enterprise to promote such shifts by providing concrete technological ideas through the integration of Knowledge media, Virtual media and Kansei media.

The third is a bit different from the first two in that it is related to artificial intelligence(AI) research where so-called form-oriented basic research has been extensively conducted. It has been trivial from the beginning that no intelligent system can function without a reasonable amount of knowledge. Nevertheless, form-oriented research has dominated AI research. Content-related activities are mainly knowledge base construction. Although huge amount of such activities have been conducted to date, they cannot be said to be "Research", since they are ad hoc, heavily domain-specific and hardly accumulatable. We do need content-oriented "Research" to make an essential contribution to intelligent system building. This is related to Knowledge media technology.

The fourth is related to system architecture issues. It is an infrastructure of building a large-scale robust systems which are often difficult to build and maintain. Typical distributed control systems include a multi-agent system in which agents collaborate with each other without a priori specification of interaction between them unlike the conventional centralized control systems. This paradigm of system design makes it easier to build a large-scale systems provided a powerful negotiation protocol.

On the basis of the background described above, we decided to employ the following four major policies to conduct the project:

(a) Open-architecture including human operators rather than closed one where a system tries to solve everything by itself: This policy is the most straightforward decision made according to the project mission of building a sophisticated human interface for plant operators. The initiative should be

taken by the human operators who are responsible for every decision. The role of systems in support of the plant operation has to be not "passive" but "active". By active support, we mean information is given to the operators not only by their demands but also system's spontaneous decision on what information is necessary and when it should be provided. Such support systems should have potentials of coping with the difficulties and such capabilities enables them to work with human operators who require an intelligent partner which could augment their capabilities. Such systems apparently have to provide human operators with useful and timely information about the plant including suspected causes with operation recommendations with justifications. For that purpose, we employ an Interface Agent which monitors the dynamic states of the plant and notifies operators of noteworthy events as well as conventional messages to operators. It complies with the paradigm shifts (1) and (2).

(b) Introduction of ontological engineering to avoid ad hoc knowledge representation and utilization: Conventional AI research has concentrated on formal issues such as logic, knowledge representation, formal reasoning, etc. However, the real world problems require sophisticated treatment of knowledge, that is, to cope with the issues of how to make the knowledge base construction principled and how to make the knowledge bases reusable. Ontological engineering is expected to give a solution to this. It complies with the paradigm shift (3).

(c) Multi-agent architecture rather than centralized one: Such a system that we plan to build will necessarily become a large-scale system which is hard to develop, control and maintain. The multi-agent architecture with distributed control has many merits provided we come up with powerful collaboration protocol and vocabulary/concepts which are two of the major research topics of the project. It complies with (4)

(d) Case-based and model-based approach rather than heuristics-based one: Unlike the conventional expert systems, we decided to employ Case-Based Reasoning: CBR for monitoring the plant and Model-Based Reasoning: MBR for understanding the plant state with prediction of plant behavior.

These polices make the system unique and different from conventional systems of similar purposes. It is not an expert system in conventional sense. The next section discusses motivation and background of the project together with project organization.

2. SPECIFIC MOTIVATIONS AND PROJECT ORGANIZATION

The current state of the art of the plant operator support

We adopted an oil refinery plant as a task domain. After general investigation and several interviews with domain experts and plant engineers, the following current situations of plant operation were revealed:

(a) Operators have little chances to acquire know-how of plant operation because of the increase of reliability of the plant.

(b) A plant has become a large black box to operators because of its growth in size and advancement of automation.

(c) The human interface available is at best a multiwindow system of DCS: Distributed Control Systems which is totally a passive information provider with an uncontrolled alarm system.

(d) The way of information provision is monotone rather than adaptive to operators and situations.

(e) Many kinds of information is available upon request, but all given is unrelated and disordered. Lack of integration.

(f) All the operator support systems are built from scratch and hence expensive.

(g) Maintenance of the support systems has become more and more difficult according to the growth of the plant in scale.

The goal of our project is apparently to resolve these difficulties. A plant should be transparent to the operators. The interface cannot be a simple thing which exists between the system and operators, rather, it should be a **mediator** which assists operators in making appropriate decisions by giving necessary and sufficient information at appropriate time based on a deep understanding of the plant itself and characteristics of human perception and cognition. Information provision should be not passive but active. Information should be integrated in the sense that all pieces of information should be controlled by a principled manner according to an explicit goal of the provision. In order to enable the interface to understand the plant, it should have a model of the plant and hence it should employ model-based reasoning. Such an interface necessarily becomes a huge system. But, it should not be difficult to build. The knowledge bases should be reusable for building similar systems later on. It requires an ontology engineering to build reusable knowledge bases. Such a system should be flexible in the cases of building and execution, which suggests a multi-agent architecture with negotiation functionality.

Project organization

On the basis of the above observations, we envision a resulting operator support system as shown in Fig. 1. It consists of several displays for specified types of information such as a virtual plant with walk-through facility, an interface agent with spoken dialog functions, sophisticated message generation based on model-based reasoning. All the activities of the interface are supported an advanced multi-agent architecture with an ontology server. The following is the overview of the topic organization set up to execute the research plan. Fig. 2 shows the block diagram of the interface system.

(1) Interface agent: IA

It is located between the human operators and the plant to monitor the plant states for them, triggers the collaborative actions among other agents and gives messages to operators in both text and speech. It is an "agent" in the following two meanings: An agent which performs monitoring job for human operators and an agent which can autonomously work with other agents. It consists of two major modules: case-based reasoning for plant monitoring and spoken language processing for human interface. The former stores cases of time series of various sensor data of the plant together with labels(normal, abnormal, etc.), diagnostic information and operations done for the situation. All the information is indexed for retrieval and later use for plant monitoring, diagnosis and operation. Techniques used in the spoken language processing include Direct Memory Access Parser: DMAP and Memory Organization Packet: MOP. Speech input and output is carried out using commercially available speech boards.

It employs no heuristics of the operators to make it free from the difficulties that conventional expert systems had. It can learn through its experience by increasing the case base.

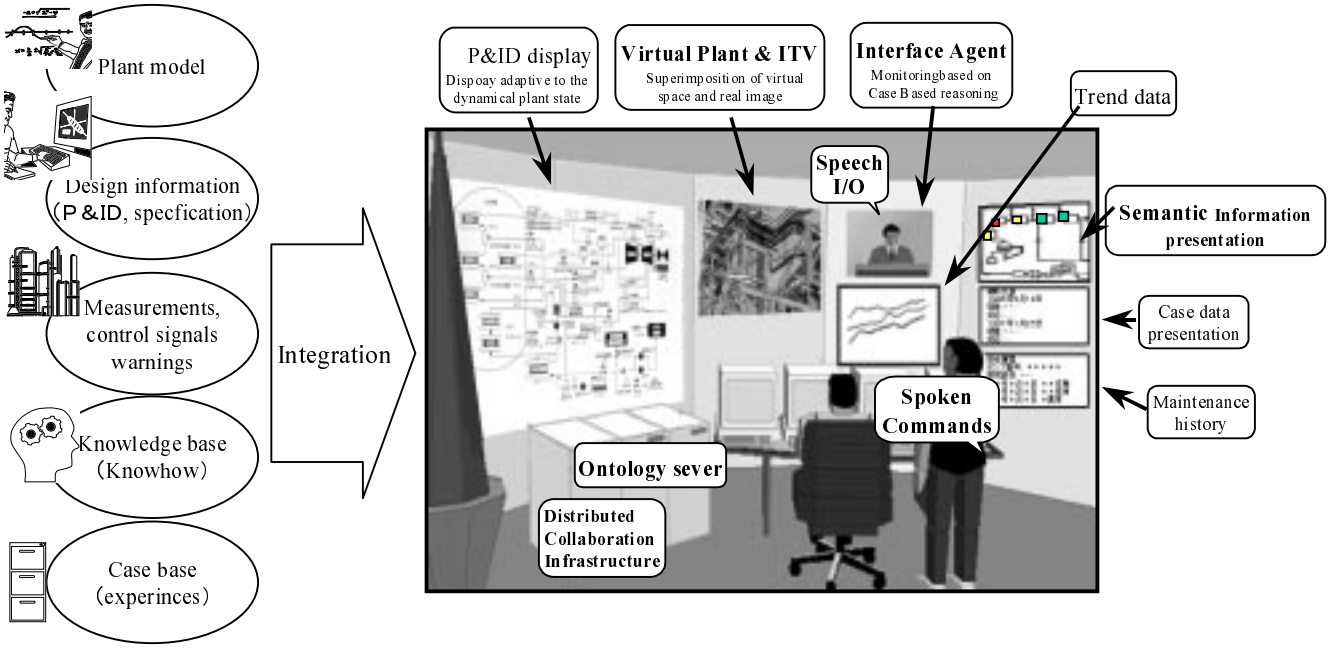

Fig. 1 An envisioned interface system

(2) Virtual plant agent: VPA
VP is responsible for realizing a 3-D virtual plant for easy access to any portion of the plant. It allows operators to inspect any sensor data for his decision making as well as virtual operation on devices. It makes the plant physically transparent by visualization. The virtual plant has been carefully designed to imitate the real plant in a good fidelity so that plant operators can enjoy highly situated cognitive activities.

(3) Semantic information presentation agent: SIA
The major functionality of SIA includes to provide semantic information of the plant based on multi-layered logical model of plant rather than raw data. Given candidates of causes of a fault from IA, it identifies the most plausible one with restoration operation for it. IA and SIA are complementary to each other in that SIA reasons about the plant using model-based methodology, while IA employs a CBR approach based only on the past experiences. When IA finds a very similar case, the decisions drawn from it might be highly reliable. On the other hand, if it cannot, CBR cannot say anything useful. Model-based approach is generally robust as far as the plant states exist within the coverage of the model built. Thus, the two compensate for each other.

(4) Ontology server: OS
Each agent has their own specialty which differs a lot from each other. It is a must for them to share minimum commitment on understanding of the domain to collaborate with each other. A solution to this is an ontology. An ontology is a shared theory/system of concepts of the target world to specify its conceptualization. The concepts and vocabulary in an ontology are used as building blocks of the operational model of the world. The ontology server, OS stores an ontology developed for the target oil-refinery plant and serves it with agents. All agents necessarily commit to the shared understanding by using the ontology and hence they can maintain fluent communication. Further, OS has a function of message generation in terms of appropriate words. OS is always asked to generate a message by IA who wants to say something to the operators. It sends OS stylized meaning representation which should be converted into natural

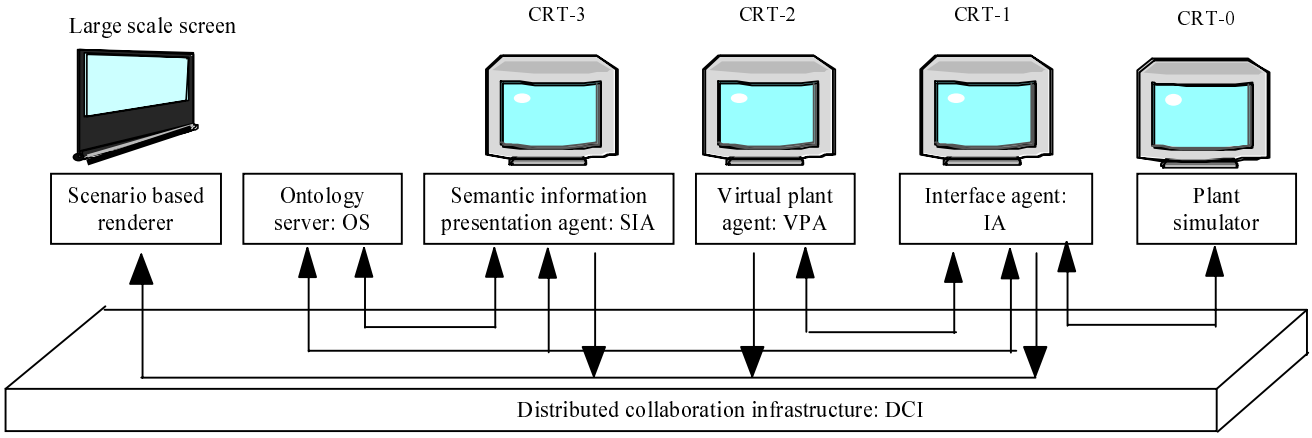

Fig. 2 Block diagram of the entire interface systems.

language, Japanese in our case. The word selection is done consulting the model of the target plant and definitions of terms in the ontology.

(5) Distributed collaboration infrastructure: DCI
The infrastructure performs the functions of a communication channel and a negotiation facilitator for the communication among agents. Needless to say, DCI and OS are compliant with the above policies. They make the whole system robust in runtime and easy to maintain thanks to its sophisticated negotiation protocol. An agent only has to know what jobs they want to get done and send a message using the local terms which have been told OS in advance without knowing who could do the job. The unique point of DCI is that it introduces **Proxy agents** (in our system they are called **Task agents**) which do the negotiation among them for the agents they represent. Therefore, addition and modification of any agent is relatively easy.

(6) Plant simulator
The system requires real time plant operation data. At an actual plant, DCS (Distributed Control System) is installed and manipulates the plant data every second. In this project, a real time dynamic simulator instead of a real plant is installed, using the mathematical model representing a Topper (Crude Distillation Unit). This model consists of the following major equipment and instrumentation.

Desalter
Preheat Train (19 Heat Exchangers)
Preflush Drum
Crude Furnace
Crude Tower (42 Trays)
3 Side Strippers
37 PID Controllers
18 Pumps
132 Analog Indicators and Monitors
93 Field Valves

A set of tuning parameters such as PID and alarm range can be changed arbitrarily. This model can produce emergency operations such as pump degradation and failure, tube leakage, valve stuck and controller failure etc. After the set values of controller are changed or malfunctions are set, the operating condition will be changed. This model can calculate the changing data in every second and send the data to Interface Agent in every 2 seconds.

3. OVERVIEW OF EACH AGENT

Interface Agent

In process plant operation, an operator reads time series data, senses happenings, identifies a plant state, and foresees a future state. A cognitive agent that has no knowledge but competence of situation awareness would be able to sense anomaly on behalf of an operator, inform happenings to the operator, and help him/her identify the current plant state. The cognitive agent that learns from experience would be able to grow its capability being taught by operators, starting without knowledge just same as a human novice would become an expert.

As the framework of agent memory and inference mechanism, we adopted case-based reasoning (CBR). Sensation of anomaly is built on Memory Organization Package (MOP). In the memory, snapshots of time series plant data make nodes and all of them are classified in version space on each item of plant data or an attribute. Thus, exact same snapshots or same occurrences in plant states fall into an exact same node in the memory, and similar snapshots or states fall into the neighborhood in the version space.

The demonstration using the simulator of an oil refinery plant proved that the agent is helpful in plant operation.

Virtual Plant Agent

The VPA enables the operators to recognize the behavior and states of the plant intuitively providing the following functions.

(1) To present information they want in a 3D virtual space.
(2) To enable operators to walk through the plant for inspection.
(3) To control the amount of information represented so as to be neither too much nor too little(in this case, it is visual information).
(4) To enable an interactive information presentation.
(5) To help operators properly recognize the process state of the plant.

VPA has been implemented to realize these functions with tight interaction with Interface Agent.

Architecture of VPA

VPA displays various information in 3D virtual space by commands issued by Interface Agent(IA). Figure 3 shows the block diagram of VPA. It decodes a command from IA and its content controls VR Module. On the other hand, concerning message presentation, VPA creates another window on the VR Display and inserts character sequences in it. And the event via Display and VR Control from VR Module caused by the operator's mouse manipulation is encoded to the command and then the command is transferred to IA. Thus, there are two types of interactive interface by both commands from IA and operator's manipulation.

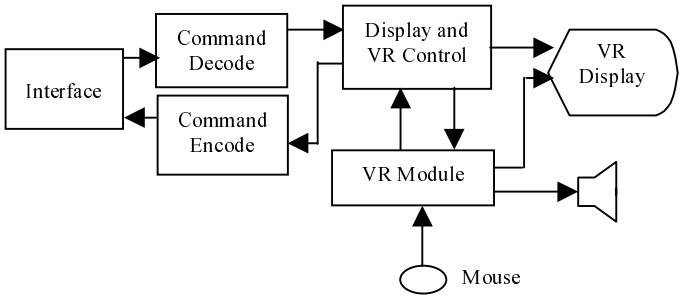

Fig. 3 Architecture of virtual palnt agent.

Semantic information presentation agent

In consideration of the human characteristics in an emergency situation, an information display interface system should efficiently indicate the necessary information

(1) to make operators not to be trapped in a cognitive narrow path,
(2) to make up operators' cognitive resources (awareness and knowledge about plant) and to support the efficient usage of the resources, and
(3) to support to organize an accurate mental model of the plant situation through multi-modal information channel.

In order to meet the requirements mentioned above, an intelligent agent called Semantic Information presentation Agent(SIA) is

developed to provide diagnostic information from multiple viewpoints. The basic approaches of the SIA are

(h) modeling and usage of designers' intentions of the plant by a functional modeling framework in addition to the information of plant behaviors, structure, and operation procedures,
(i) generation of diagnostic information such as identification of anomaly cause, derivation of possible countermeasures, and evaluation of the effects of a countermeasure by model-based reasoning, and
(j) application of adaptive interface concept[1] to display the diagnostic information corresponding to the information category to which operators should pay attention.

Functional information of a plant will help operators understand overall plant situation and the necessity of a recommended countermeasure by the operation manuals or the SIA because it expresses the reason and background of the existence of components. The relevant functions and components with plant anomaly or a recommended countermeasure are also displayed in a graphical way which operators can quickly understand the information as a pattern. **These advantages will lead the decrease of mis-operation and can make operators to monitor effectively the transition of plant condition after taking a** countermeasure**.**

The SIA is composed of (a) plant model, (b) identification sub-system of anomaly cause from a set of probable causes given by the interface agent: IA through distributed collaboration infrastructure: DCI**, (c) derivation sub-system of possible** countermeasures**, (d) evaluation sub-system of the effects of a** countermeasure**, and (e) display control sub-system to flexibly indicate diagnostic information depending on the information to which operators pay attention. The plant model is based on a functional modeling framework, Multi-level Flow Modelling (MFM)[2], to express intentional information of plants, that is, goals and functions as well as structural information. The plant model also includes the information such as qualitative relations among goals and functions, operator actions and component behavior at the happening of an anomaly. The qualitative reasoning technique is applied to develop the sub-systems (b) and (c). An intelligent modular simulation technique[3] is adopted to generate a simulation code to predict future trend of plant behavior in the sub-system (d). The current status of the SIA implementation is as follows:**

(1) a graphical interface system is developed to support the construction of a plant model[4],
(2) algorithms for sub-systems (b) and (c) are proposed[4,5],
(3) the intelligent modular simulation technique is implemented, and
(4) a simulation model to predict plant future behavior of a oil refinery plant is developed and is now under implementation. The algorithms are applied to a simplified plant model of the oil refinery plant and are proved to offer useful diagnostic information.

Plant ontology and ontology server

An ontology is a backbone of the target world of interest[8]. In our system, several plant ontologies are stored in Ontology server: OS for use by other agents. The main purposes of OS include:

(1) To store the plant ontology for use of representing message contents
(2) To build a structural model of the target plant shared by other agents
(3) To generate natural language messages presented to operators according to appropriate word selection
(4) To generate explanation about behavior and function of sub-components specified.
(5) To provide a negotiation ontology for smooth collaboration among agents.

In general, the major contribution of OS to the whole system is to standardize concepts about the target plant as well as vocabulary used by agents. We built a plant ontology which consists of several hierarchical organizations of concepts such as operation task ontology[7], plant components & objects, and basic attributes & ordinary attribute. The key issue in design of an ontology is clear distinction between essential categories and peripheral or view-dependent concepts.

The operation task ontology consists of action done by the task performer, operators in our case, and of the role which domain objects play during the task performance. There exist two major things in the plant domain: plant components(devices) and plant objects to be processed by the devices. Such domain concepts also have role concepts like task ontology. To say precisely, many of the domain concepts are role concepts. The top level categories of ***plant object*** are ***view-independent object*** and ***view-dependent object***. The former includes LP gas, gasoline, naphtha, etc. which are categories persistent in any situation. The latter includes ***tower-head element, liquid, distillate, input, intermediate product, raw material, fuel***, etc. All are view- or context-dependent.

Another kind of domain ontology is necessary to build a model of a plant as an active artifact[6]. That is an ontology of function & behavior which is a domain-independent. We built an ontology which contains about 350 concepts which are approved by the domain experts and the coverage is around the topper(normal pressure fractionater) of a full-scale refinery plant.

The plant model is built by connecting components generated by instantiating the corresponding concepts in the ontology and made accessible from other agents. In the current implementation, the major function of OS is word selection in the message generation given to operators. In order to minimize the cognitive load of operators in message understanding, stylization of the message and careful word selection is critical. The efficient word selection is realized thanks to the plant ontology in which words are properly defined with a logical model of the target plant. The algorithm consists of two major sub-algorithms: context tracing and word generation. Use of word is highly dependent on the context which is represented as focused device in our algorithm. Under an identified context, word selection is done by deciding on either use of a default word associated with each object or generation of one according to the context.

Ontology server has been implemented in Java and Lisp. Evaluation has also been done as to message generation. Although the scale was small, eight cases were tested, the result was successful. It has an ontology building environment, called Ontology Editor, and has another function, consistency check of the model built from the ontology.

Distributed collaboration infrastructure: DCI

Distributed collaboration infrastructure: DCI is a computational framework which allows each agent, a constituent unit of the framework, processes its own tasks in a collaborative way by communication with other agents under a loose control of the framework. It enables each agent to easily expand its functions. For processing tasks in DCI environment, it is necessary for the framework to integrate functions of agents rationally for collaboration. In other words, basic software is necessary for taking partial charge a process to subsystems and collecting results processed by them. We propose an architecture that integrates functions of subsystems in DCI environment. Our architecture is based on a method of an agent technology. It is better than the conventional ones in that it is easy to incorporate existing modules to DCI environment.

Firstly, we consider a concept model for collaboration and integration. In previous work, we studied integration of software and information in a distributed environment [9]. It is possible for the model to take partial charge of a task using Task field and Task agent(Proxy agent). Task agent is a software module for communicating with other agents using a common protocol instead of an agent, and has functions of communication and collaboration between agents, and demands processing tasks to an agent. Task Field is a communication space for Task agents which is controlled by Collaboration server described below. For communication between Task agents, we use Contract Net Protocol [10] as a collaboration protocol, and KQML (Knowledge Query and Manipulation Language) [11] as a communication language. Task agents decide on a procedure and a role under the protocol. If Task agents cannot process a task in an existing Task field, the agent divides the task into several smaller sub-tasks, using related knowledge databases, and prepares a new Task field and Task agents for processing each sub-task.

Secondly, we consider mechanisms for DCI, Collaboration server and Distributed collaborative API. Collaboration server manages a collaborative process that creates and deletes Task fields and Task agents according to requests from agents in a same environment. Distributed collaborative API (Application Program Interface) is a message format for communication between an agent and its Task agent in Task field. The API sends a request message and a result message to Task agents in Task field. The API has three features, data exchange by passing objects, asynchronous communication, and management by a collaborative processing system that manages fields and Task agents. We used the software and a protocol, TCP/IP as a communication protocol for building a network, JAVA (JDK 1.1.x) as the software building program, HORB as the communication software for constructing the distributed environment. HORB [12] is ORB coded in JAVA language. In order to preserve flexibility and allow Task agents and fields to be constructed dynamically, the program for each agent is simply a class code. Using the architecture, agents can process tasks with little consideration about collaboration. Also existing programs can join in DCI only by adapting Distributed collaborative API.

We developed a prototype system by imitating a human interface system for an oil refinery plan using the architecture in DCI and evaluated using an example of information presentation by multiple agents and had a successful result.

4. CONCLUSION

We had an intermediate term demonstration in the March in 1999 where all the subsystems developed thus far were presented independently except IA and VPA using a few realistic scenarios and had a positive feedback then. The major issues to pursue include the integration all the agents into the DCI with sophisticated a collaboration protocol as well as full evaluation.

What is lacking in our project is topics related to human factors which, from this year, we plan to seriously investigate and incorporate the results into the way of information presentation.

ACKNOWLEDGMENTS

This paper was prepared under an Entrustment Contract with the Laboratories of Image Information Science and Technology (LIST) from the New Energy and Industrial Technology Development Organization (NEDO) in concern with the Human Media Research and Development Project under the Industrial Science and Technology Frontier (ISTF) program of the Ministry of International Trade and Industry (MITI) of Japan.

5. REFERENCES

[1] Rouse, W. B.: Design for Success, John Wiley & Sons., 1991.

[2] Lind, M.: Applied Artificial Intelligence, Vol. 8, No. 2, 1994, 259-283.

[3] Gofuku, A. et al. :Proc. IJCAI Third Workshop on Engineering Problems for Qualitative Reasoning, 1997, 19-27.

[4] Gofuku, A, et al.: Proc. IJCAI Fourth Workshop on Engineering Problems for Qualitative Reasoning,1999,to appear.

[5] Gofuku, A. et al.: Proc. Fifth Int. Workshop on Functional Modeling of Complex Technical Systems, 1998, 87-97.

[6] Y. Kitamura and R. Mizoguchi, Meta-Functions of Artifacts, The Thirteenth International Workshop on Qualitative Reasoning (QR-99), Scotland, June 6-9 1999.

[7] M.Ikeda et al., An ontology for building a conceptual problem solving model, Workshop Note of Applications of ontologies and problem solving methods, ECAI98, 1998.

[8] Riichiro Mizoguchi, A step towards ontological engineerign, National Conference on AI of JSAI, AI-L13, 1998.

[9] Tokumoto, S., and Maenaka, A.: An Architecture for Cooperative Processing using Hierarchical Cooperative Processing Fields. Proceedings of the 12th Annual Conference of JSAI, 580-582, 1998.

[10] Smith, R. D.: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computers, Vol. C-29, No.12, 1104-1113, 1980.

[11] Finin, T., et al..: DRAFT Specification of the KQML Agent-Communication Language plus example agent policies and architectures. The DARPA Knowledge Sharing Initiative External Interfaces Working Group, 1993.

[12] Hirano, S., HORB: Distributed Execution of Java Programs. World Wide Computing and Its Applications, 1997.