Agent-Based Design Model of Adaptive Distributed Systems

Article *in* Applied Intelligence · July 1998

DOI: 10.1023/A:1008299131268 · Source: DBLP

5 authors, including:

Shigeru Fujita
Chiba Institute of Technology
43 PUBLICATIONS **261** CITATIONS

SEE PROFILE

Hideki Hara
Chiba Institute of Technology
14 PUBLICATIONS **166** CITATIONS

SEE PROFILE

Kenji Sugawara
Chiba Institute of Technology
120 PUBLICATIONS **938** CITATIONS

SEE PROFILE

Tetsuo Kinoshita
Tohoku University
254 PUBLICATIONS **1,511** CITATIONS

SEE PROFILE

Agent-based Design Model of Adaptive Distributed Systems

SHIGERU FUJITA, HIDEKI HARA, KENJI SUGAWARA

fujita, hara, suga@suga.cs.it-chiba.ac.jp

Department of Computer Science, Chiba Institute of Technology, 2-17-1, Tudanuma, Narashino, 275 JAPAN

TETSUO KINOSHITA, NORIO SHIRATORI

kino, norio@shiratori.riec.tohoku.ac.jp

Electrical Communication, Tohoku University, 1-1-1, Katahira, Aobaku, Sendai, 980, JAPAN

Received ??; Revised ??

Editors: ??

Keywords: adaptive distributed systems, cooperation protocol, design model

Abstract. A next generation distributed system is expected to adapt to various changes of both the users' requirements and the operational conditions of environment where the distributed system operates. The aim of our research is to establish a new design model of an adaptive distributed system (ADS) to deal with various changes occurred in the system environment. In this paper, we propose an agent-based architecture of the ADS based on the agent-based computing paradigm. Then, we implement a prototype of the ADS with respect to video-conferencing applications and also evaluate the adaptive functions of the ADS realized on the basis of the proposed architecture.

1. Introduction

The next generation distributed systems have to provide various services for the users living in the networked information spaces based on the large-scale information networks. In the real world, the information networks confront with various perturbations of their functions which are caused by the changes of both the users' requirements and the operational conditions of the networks. As a result, the quality of service of the distributed systems is changed and degraded together with the perturbations of the information networks. Recently, the studies aiming this problem have been started in the field of both the advanced information networks [3, 5, 11,13,15,16,19] and the distributed multimedia applications [2,11,12,20,21,22]. In order to provide the user-oriented, intelligent and stable services for the users, a distributed system should have a mechanism to maintain the quality of service against various changes observed in the operational environment. With this mechanism, a distributed system has an adaptive capability to deal with the fluctuating operational environment, and we call such a system an adaptable distributed system.

In order to establish a design model of an adaptive distributed system (ADS), we define a notion of adaptation from a view point of the changes of both the users' requirements and the operational conditions of the systems' environment, and propose the adaptation functions which provide the adaptive capabilities for a distributed system. Then, we propose an agent-based architecture of an ADS by which the adaptation functions are realized by using the agent-based computing technologies [1,11,17,,23]. Furthermore, in this paper, we design and implement a prototype of the ADS and also evaluate the adaptation functions of the ADS with respect to the proposed

notion of adaptation of distributed system. In chapter 2, we introduce a notion of adaptation of distributed system, and propose five kinds of the adaptation functions of the ADS. An example is also given to depict the adaptive behavior of the ADS. In chapter 3, an agent-based architecture of the ADS is proposed together with four kinds of the cooperation protocols among the agents of the ADS. In chapter 4, an adaptive video-conferencing system is designed and implemented as a prototype of the ADS based on an agent-oriented programming environment called ADIPS framework [6,9] which has been developed by the authors. In chapter 5, the experimental results of the prototype are demonstrated to show that the adaptation functions of the ADS can make a video-conferencing system adaptive with respect to the changes of both the users' requirements and the operational conditions of system's environment.

2. Design Concept of Adaptive Distributed Systems

2.1. Notion of Adaptation in Distributed System

A distributed system consists of various distributed computer systems and the information networks which are called a platform of the distributed system. According to the requests given by the users, the services of the distributed system are realized by using the functions of the platform and provided for users. From the view point of the configuration of the services of the distributed system, the following changes can be considered at the run time of the system, i.e.,
(V1) change of the quality of service (QoS) of a platform: due to the growth of the communication traffic or the computational load of the platform, the QoS provided for the users are degraded as the computational resources of the platform are decreased.
(V2) variation of the required services: various kinds of the services are realized in accordance with the user's requests in which the quality of the required services (the required QoS) are spcfified.
(V3) change of the users' requirements: due to change of both the required services and the provided QoS of the platform, the users' requirements can easily be shifted to another one.

A notion of adaptation is expressed as a system's capability to deal with these changes, i.e., the adaptation of distributed system is that the distributed system can modify its structure and functions against various changes of the operational environment observed at the run time, to realize and maintain the required services which maximally satisfies the required QoS under the provided QoS of a platform at that time. A distributed system with such a capability is called an adaptive distributed system (ADS), in this paper.

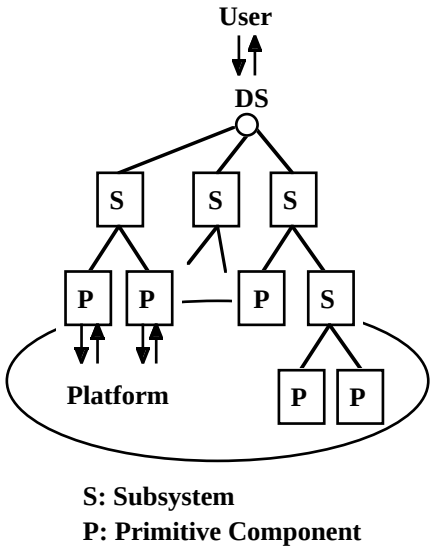

Fig.1 Structure of Distributed System

2.2 Design Model of ADS

A distributed system, in general, consists of various functional components located on a platform, as shown in Fig.1. Here, a component which consists of the other components is called a subsystem, and a component which does not have any other component is called a primitive component. Various changes will take place in both the user and the platform as explained in section 2.1. According to the notion of adaptation of distributed system, the following three methods are proposed to adapt the distributed system to its perturbed system's environment.

(M1) Tuning: Each component of a distributed system has the operational parameters to change its functional characteristics and behavior. To deal with a slight change of the operational environmen, a suitable component is selected and the operational parameters of the component are modified.

(M2) Replacement: When a function of a distributed system has been altered by the changes of the system's environment, a component which realizes the function is identified and replaced with a new component.

(M3) Reorganization: To deal with a big change of the system's environment, a subsystem of a distributed system is redesigned and reorganized into a new subsystem which will fit the whole system to the altered environment.

An adaptive distributed system (ADS) is defined as a distributed system with the adaptation functions based on the above three methods. A design model of the ADS is depicted in Fig.2.

An ADS consists of a distributed system (DS), a component repository (CR), and the five kinds of the functions which make the DS adaptive. The DS is composed of various components as explained in section 2.1, and the CR manages the components to be used in the DS. The CR can also select the components based on the requests sent from other modules of the ADS. The following functions are the essential modules of the ADS.

(1) Request Aquisition Function (RAF): receives and analyses the user's requests to detect the changes of the user's requirements.

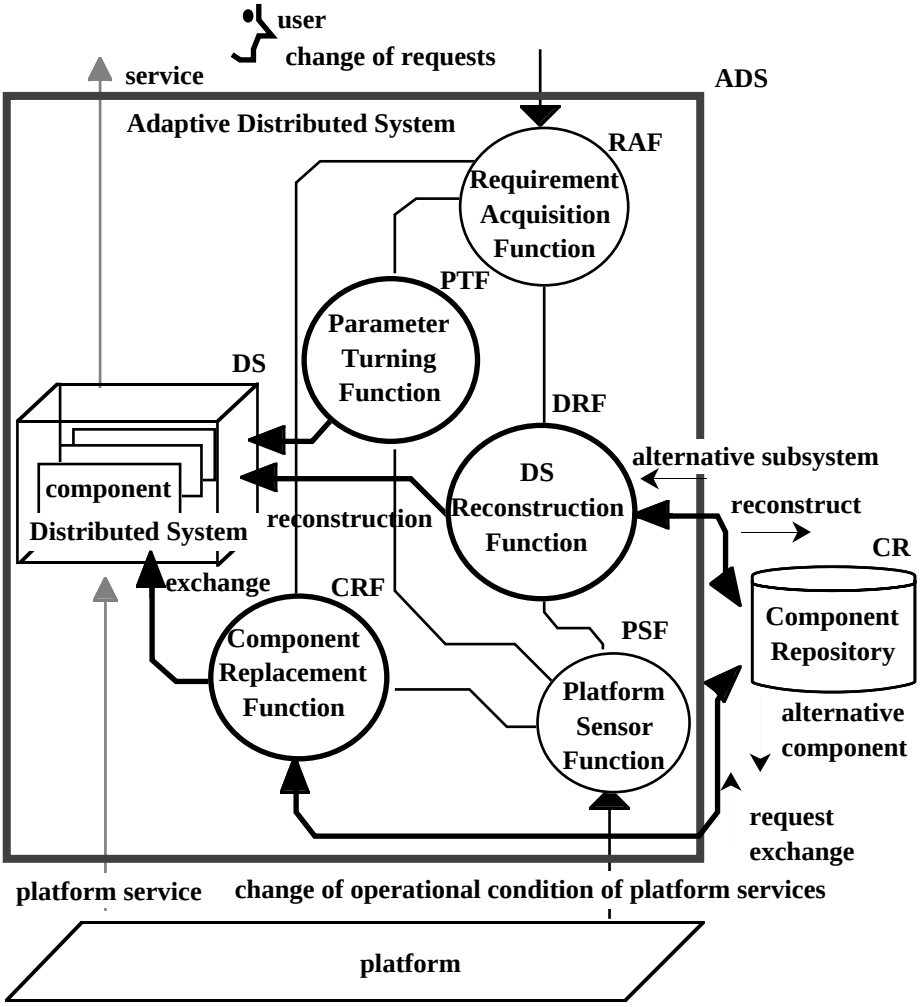

Fig.2. Design Model of Adaptive Distributed system

(2) Platform Sensor Function (PSF): monitors the operational environment of the platform and sends the reports to other functions.

(3) Parameter Tuning Function (PTF): is an adaptation function based on the adaptation method M1. Receiving the reports on the changes from the

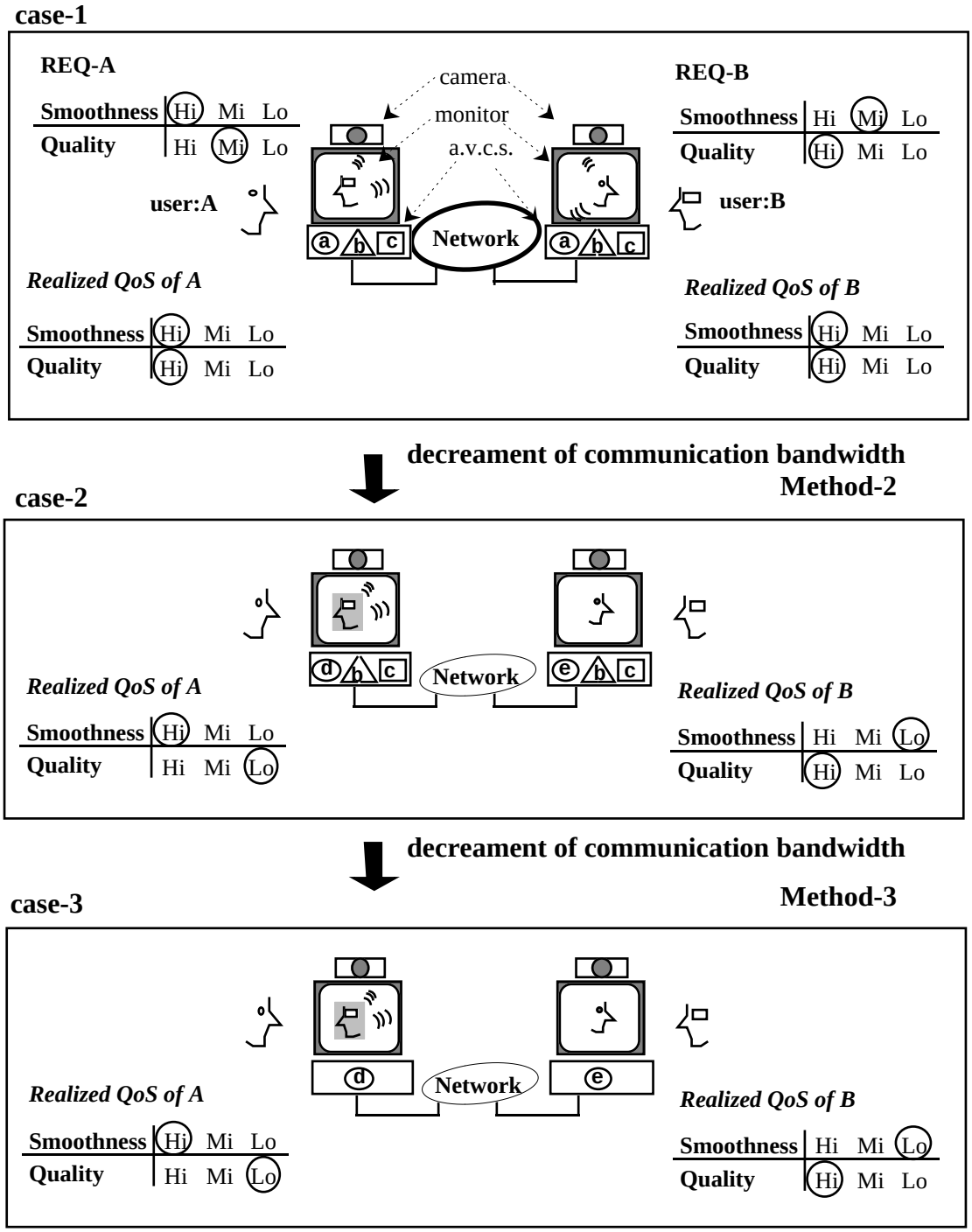

Fig.3. Example of Adaption

RAF and the PSF, the PTF makes a plan to find a component of the DS and tune the parameters of the selected component.

(4) Component Replacement Function (CRF): is an adaptation function based on the adaptation method M2. According to the reports from the RAF and the PSF, the CRF makes a plan to replace a component of the DS with a new component selected by the CR, and then, the CRF executes this plan.

(5) DS Reconstruction Function (DRF): is an adaptation function to deal with a big change of the operational environment based on the adaptation method M3. As same as the CRF, receiving the reports from the RAF and PSF, the DRF determines a subsystem to be redesigned and makes a plan to build a new subsystem. According to the plan, the DRF selects the components from the CR, combines them into a new subsystem, and reorganize the whole ADS.

2.3 Example of Adaptation of ADS

Let's consider a two-persons video-conferencing system as a typical example to grasp the idea of the ADS. In this system, for instance, the users, i.e., User-A and User-B, state their requests on the moving picture images and specify the QoS parameters of the video service components as follows, i.e.,

(REQ-A) User-A prefers the high smoothness to the quality,

(REQ-B) User-B prefers the high quality to the smoothness.

According these requests, a two-persons video-conferencing system is realized by two subsystems "Sa" and "Sb", using the same components, i.e., "a" is a full-color video service component, "b" is an audio service component, and "c" is a common boad service, as shown in case-1 of Fig.3. In the initial stage, the QoS of the platform is good enough to realize the users' requirements, hence, the high smothness and high quality of video service are attained simultaneously.

At the run time, due to the decrement of the communication bandwidth of networks, for instance, the QoS of the platform is degraded and the required QoS cannot be satisfied. In such a situation, according to the degree of decrement of the bandwith, the following adaptation plans may be considered, i.e.,

(1) Plan-1 for slight decrement of bandwidth: Applying the proposed method M1, the QoS parameters of the component "a" of both "Sa" and "Sb" are modified to keep the respective users' preferences. For instance, the PTF makes a plan that the smoothness remains high and the quality becomes low in "Sa", and the smoothness becoms low and the quality remains high in "Sb".

(2) Plan-2 for large decrement of bandwidth: When the PTF cannot make any reasonable plan, the CRF is activated by the method M2. In order to maintain the high smoothness in "Sa", for instance, the CRF makes a plan that a component "a" of "Sa" is replaced with a new component "d" which provides a monochrome video service, and a component "b" of "Sb" is replaced with a component "e" which provides a high quality still image transmission service, as shown in case-2 of Fig.3.

(3) Plan-3 for more larger decrement of bandwidth: If the CRF cannot find the suitable components to be changed, the DRF is activated by the method M3. For instance, the DRF makes a plan to replace "Sa" with a new subsystem "Sa*" which has only one component, e.g., a 256-color video service, to deal with the large degradation of the QoS of the platform, as shown in case3 of Fig.3.

As explained above, the video-conferencing system with the adaptation functions can change itself to maintain the reasonable serives against the changes of the platform based on the proposed adaptation methods. Such an adaptive video-conferencing system can also adapt to the changes of the user's requirements in the same manner.

3. Agent-based Architecture for Adaptive Distributed Systems

3.1 Overview of an Agent-based Adaptive Distributed Systems

A design model of Agent-based Adaptive Distributed Systems (AADS) is shown in Fig. 4, which is modeled based on a design model of the ADS proposed in the chapter 2. The AADS consists of four kinds of agents and the cooperation protocols, described as follows:

AADS=<SA, AMA, UA, PSA, PTL > where

the SA is a multiagent system which provides the services of theobjective DS for the users cooperatively, the AMA is an ADS Management Agent which realizes

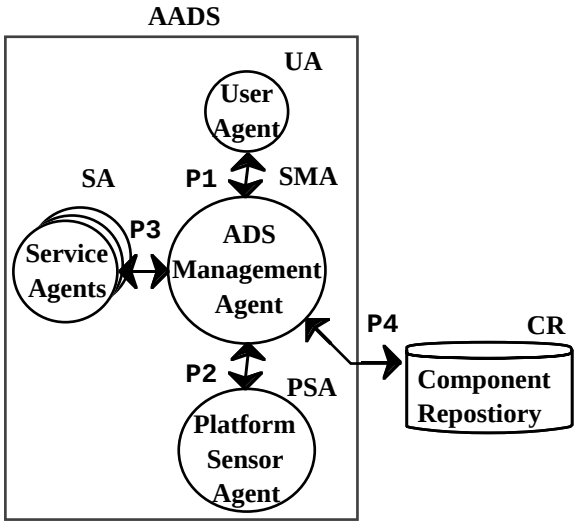

Fig. 4. Agent-oriented Adaptive Distributed Sytem

the functions of the PTF, CRF, DRF shown in Fig,2, the UA is a User Agent which realizes the RAF, the PSA is a Platform Sensor Agent which realizes the PSF. The details of functions of the agents are designed in section 3.2. On the other hand, as shown in Fig.4, the PTL consists of the following four classes of the cooperation protocols, i.e.,

PTL = <P1, P2, P3, P4>

The details of P1, P2, P3, P4 are designed in section 3.3. The implementation of the AADS using the ADIPS framework explained in chapter 4.

3.2 Design of Agents of AADS

(1) Service Agent: SA

A Service Agent (SA) is a multiagent system composed of the distributed agents in a platform to deal with the required tasks cooperatively, such as video conferencing functions, information retrieval functions for data bases, file access functions and so on. The agents are organized according to a functional hierarchy of the DS as shown in Fig.1. A leaf agent in the hierarchy tree deals with a required task using the computational processes controlled by this agent. On the other hand, an agent within the hierarchy tree controls the leaf agents and the other agents to deal with a task of a subsystem of a DS.

Table 1. Knowledge of Service Agent

knowledge	speficications of the functionality which the agent can provide
PROCE	decompose the required task for innernode agents
TUNE	tune the QoS of each function controlling its computational processes or the other agents
REP-SA	report the operational condition of the agent to AMA
RECOV	diagnose and recover itself when a fault has happend

Table.1 summarizes knowledge of the agents belonging to the organization of the SA. The categories of knowledge will be applied when a corresponding event happens in the SA or when a message comes from other agents based on the protocol P3. The ADIPS framework provides a language to describe knowledge of SA as explained in section 4.1.

To tune the QoS, each agent can control the parameters of computational processes or the other agents using knowledge TUNE of Table 1. The services of an AADS are realized as an organization of agents designed to satisfy the user's requirement using knowledge TASK and PROC. Furthermore, the organization of agents can be changed to adapt itself to the change of the operational conditions of the system's environment using knowledge REP-SA and RECOV. The reorganization of the SA is driven by the AMA based on the protocol P1.

(2) ADS Management Agent: AMA

The categories of knowledge of the AMA are summarized in Table.2. An ADS Management Agent AMA is driven by the messages from the UA and the AMA commits to the change of the user's requests using knowledge COOP-UA and DECISION. Then, the AMA selects an adaptation method from the proposed method M1, M2, M3, defined in section 2.2, and sends a command for adaptation to the SA by using knowledge M1, M2, M3. According to the protocol P4, the new agents are selected from a Component Repository CR and the unsuitable agents of the SA are replaced with the new agents using

Table 2: Knowledge of ADM Management Agent

knowledge	*speficications of the functionality which the agent can provide*
COOP-UA	cooperate with UA to commit user's requirement
COOP-PSA	cooperate with the PSA to contract the terms of sensing and identify the change of the platform
IDE-SA	identify the reported event occuered in SA
DECISION	decide the method to adapt the SA based on the analysis of the COOP-UA, COOP-PA and REP-SA
M1, M2, M3	give the detailed direction to the SA to adapt to the identified change, driven by the DECISION
COMPO	retrieve components and contract them to join the SA to adapt the change

knowledge COMPO.

According to the protocol P2, the AMA deals with the messages from the Platform Sensor Agent PSA and detects the change of the pratform's QoS by using knowledge COOP-PSA.

When an illegal event happens in the SA, such an event is reported to the AMA based on the protocol P3. And then, the AMA analyses the event using knowledge IDE-SA and selects an adaptation method using knowledge DECISION.

(3) User Agent: UA

In Table.3, the categories of knowledge of the UA is summarized. The UA acquires the user's requests via a dialog box displaied and controlled using knowledge DILOG, and transforms the ambiguous and incomplete requests to the formal descriptions of the protocol P1, using knowledge TRANS. Applying knowledge COOP-AMA and COMMIT, the UA informs the users that the QoS of the SA is changed due to the change of the platform's QoS.

Table 3: Knowledge of User Agent

knowledge	*speficications of the functionality which the agent can provide*
DIALOG	control dialog systems with users to get the agent can provide
TRANS	transform the ambiguous and imcomplete representaion of user's requrement to the formal description for agents
COOP-AMA	cooperate with AMA to inform the user's requirements of the change of them, and to recieve AMA's request of change of the QoS due to change of platform
COMMIT	to ask an user of permission to change the QoS proposed by AMA

(4) Platform Sensor Agent: PSA

The categories of knowledge of a PSA is shown in Table.4. According to the protocol P2, the PSA make a contract with the AMA to report the events observed in the platform using knowledge TERMS and CONTRACT. The monitoring of the platform is driven by knowledge SENSE, and a report of the state of the platform is sent to the AMA based on the contract using knowledge REP-PSA.

3.3 Design of Cooperation Protocols of AADS

(1) Protocol P1

The protocol P1 is utilized for the cooperation between the UA and the AMA to acquire and commit the user's requests. The messages defined in the protocol P1 are summarized in Table.5. The massage format is defined in a similar form of the KQML as explained in section 4.1. A message sent to an agent is processed using the corresponding knowledge of the agent designed in the previous section.

An example of a message sequence of the protocol P1 which determines the QoS of a user, is shown in Fig.5. The UA acquires a request from a user using knowledge DIALOG and TRANS of Table.3, and sends a message "request-service" to the AMA using knowledge COOP-AMA to request the reasonable QoS. The AMA deals with the message using

knowledge COOP-UA and recognizes that the QoS cannot be realized by the current organization of the SA at that time using knowledge DECISION. Next, the AMA sent a message "counter-request" to the UA in order to decrease the QoS of the original request.

On the other hand, the UA proposes a new QoS which may be acceptable for the user, and commits it to the user. And then, the UA sends a message "counter-request" to the AMA. In this example, the AMA accepts the counter request and a message "accept-service" is sent to the UA. As a result, the acceptable QoS is established.

Table 4: Knowledge of Platform Sensor Agents

knowledge	*speficications of the functionality which the agent can provide*
TERMS	specification of terms of the PSA can observe
CONTRACT	contract terms to report with AMA
SENSE	observe the platform controlling the sensors
RE-PSA	report the state and the events of the platform to the the AMA

Table 5: Messages for the protocol P1

message	*function*
request-service	UA requests the specified service for AMA
accept-service	AMA accepts the requiremetn of the service from UA
refuse-service	AMA refuses the requirement of the service from UA
counter-requst	An Agent propose the counter request to another agent

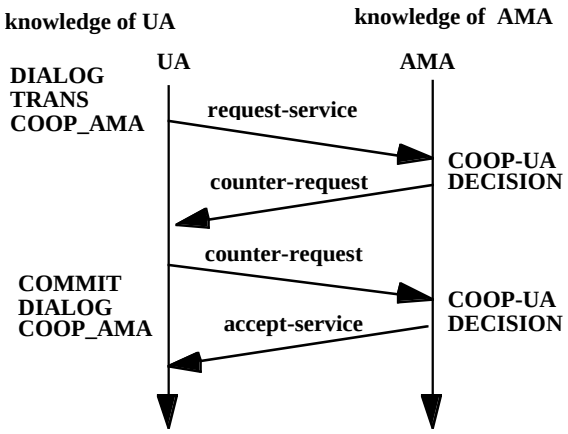

Fig. 5. Example of the protocol P1

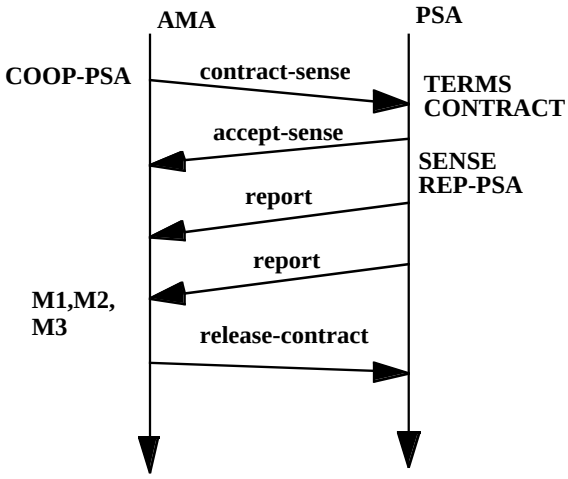

Fig.6. Example(1) of the protocol P2

(2) Protocol P2

The protocol P2 is utilized for the cooperation between the PSA and the AMA to make a contract to sensing the platform with the PSA and also to request the specified information of the platform. The messages of the protocol P2 are shown in Table.6.

An example to make a contract of detecting an event occured in the platform based on the protocol P2 is shown in Fig.6. First, the PSA accepts a contract message using knowledge TERMS and CONTRACT. According to the contract, the PSA starts the

monitoring task using knowledge SENSE. If contracted event is detected, then the PSA reports the event using knowledge REPORT. In Fig.7, an another example that the AMA requests the sensing information of the platform and the PSA returns it to the AMA is depicted.

Table 6: Messages for the protocol P2

message	*function*
contract-sense	AMA requests a contract of sensing the platform with PSA
accept-sense	PSA accepts the contract requested by AMA
report	PSA reports of sensed data or events contracted with AMA
release-contract	AMA releases the contract with PSA
request-sense	AMA requests the specified data of the platform

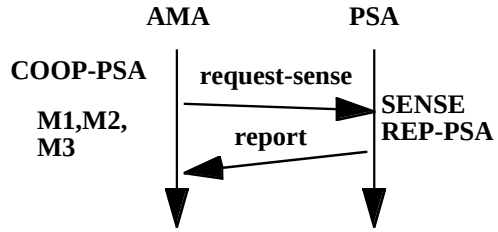

Fig.7. Example(2) of the protocol P2

(3) Protocol P3

The protocol P3 is utilized for the cooperation between the SA and the AMA to adapt the SA in accordance with the changes of the operational conditions of the system's environment. The messages of the protocol P3 are summarized in Table.7. The primitive messages such as "accept", "refusal" and "counter-request" are also available in the protocol P3.

Let's show an example of a message sequence of the protocol P3 to modify an organization of the SA, in Fig.8. The SA sends a message "report-sa-event"

Table 7: Message fro the protocol P3

message	*function*
tuning-service	AMA requires to tune the QoS of SA
replace	AMA requires to replace a component of SA to another component generated from the CR
restructure	AMA requires to reconstruct subsystem of SA with new specification of the subsystem designed in CR
report-sa-event	SA-reports an event occured in itself to request a direction from AMA
exchange	AMA sends a component (a service agent) or subsystem to be exchanged to SA
termination	AMA directions to SA to terminate

to the AMA to tell that an irregular event is observed in the SA using knowledge REP-SA. According to the message, the AMA decides to apply the adaptation method M3 and issues a command message "replace" to replace an existing agent with a new agent. Due to this message, the SA creates and installs a new agent into the organization of agents. In the case of Fig.8, an agent which controls the other agents of the organization, checks if the new agent is appropreate for the adaptation using knowledge RECOV, and finds that the irregular situation is not still recovered. After receiving a message "refusal" sent from the SA, the AMA proposes an another adaptation method for the SA by sending a message "restructure".

(4) Protocol P4

The protocol P4 is utilized for the cooperation between the CR and the AMA to select and generate the agents which will be the component agents and the subsystems of the SA. The messages of the protocol P4 are summarized in Table.8. As same as the protocol P4, the primitive messages such as "accept", "refusal" and "counter-request" are also available in the protocol

P4.

In Fig.9, an example of a message sequence of the protocol P4 to generate a component agent from the CR is depicted. The AMA sends a message "specification" which contains information of the required agent, to the CR using knowledge COMPO. Using this specification, the CR retrieves the candidates of the agents and extracts the design information of these agents which are returned to the AMA. According to information sent from the CR, the AMA selects the most suitable agent to change the organization of agents of the SA and send a message "select" to the CR. And then, the CR generates the required agent and sends a message "result" to the AMA. As a result, a new agent is instantiated and utilized as a member of the SA.

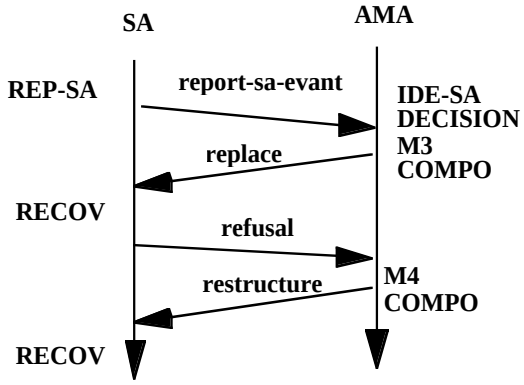

Fig.8. Example of the protocol P3

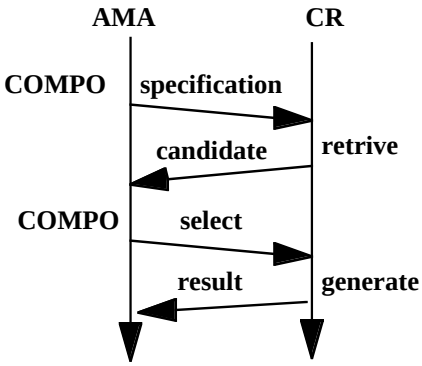

Fig.9. Example of the protocol P9

Table 8 Message for the protocol P4

message	*function*
specification	AMA sends a specification of a required component to CR
candidate	CR sends a specifications of candidate components to AMA
select	AMA sends to an identifire or an acceptable component to CR
result	CR sends to a selected component or a subsystem to AMA

4. Implementation of Adaptive Distributed System

4.1 ADIPS Framework

An experimental system of an Agent-based Adaptive Distributed System (AADS) has been developed using the ADIPS framework which supports to develop Agent-based Distributed Information Processing Systems[6,9]. The ADIPS framework is shown in Fig.10, which has two subsystems: 1) ADIPS workspace which is an operational environment of the agents and 2) ADIPS repository which is a database of the class agents. A class agent is instantiated to an instance agent which operates in the ADIPS workspace. These subsystems are implemented as a distributed system on networked workstations.

The agents in the ADIPS workspace make an organization to give sevices for users using the extended contract net protocol provided in the ADIPS framework for agents to cooperate each other, as shown in Fig.11. The extension from the Contract Net Protocol [18] is illustrated in the bold character in Fig.11.

The archtecture of the ADIPS agent, as shown in Fig.12, consists of three parts, i.e,

ADIPS Agent = < CM, PM, DK> where,

CM is a cooperation mechanism using the cooperation protocol designed based on the KQML [4] and the extended contract net protocol. The PM is a task

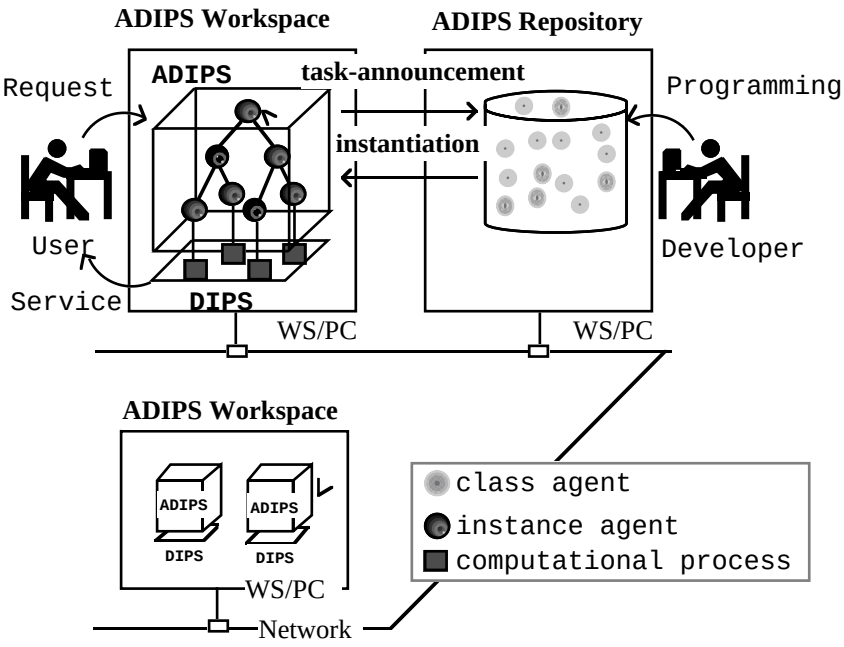

Fig.10 ADIPS Framework

```
        message := header address originator text tailaer
           text := [cctext] | pstext
         pstext := task-announcement
                 |  ...
                 |  information-message
                 |  reject-award
                 |  re-organization
                 |  stop-release
                 |  stop-request
                 |  dissolve
                 |  request-action
                 |  counter-request
   reject-award := REJECT-AWARD [name] {reject-justification}*
re-organization := RE-ORG-MSG [name] {organization-specification}*
   stop-request := STOP-REQUEST [name] {request-justification}*
   stop-release := STOP-RELEASE [name] {release-justification}*
       dissolve := DISSOLVE [name] {dissolve-justification}*
 request-action := REQUEST-ACTION [name] {action-description}*
counter-request := COUNTER-REQUEST [name] [message] {action-description}
```

Fig.11 Extended contract net protocol and cooperation protocols specification

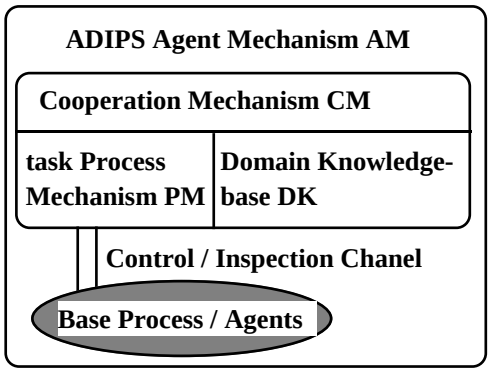

Fig.12. ADIPS agent architecture

process to deal with the specified task, such as video conference system program. The task process mechanism PM also conrtols its dominating agents to change the decomposed tasks on them. The DK is a mechanism to manage domain knowledge of each agent representing its abilities of the task perocess mechanism, know-how to realize the functions using the task process mechanism and the cooperation mechanism, know-how to adjust the functions according to the situation.

The structure of the domain knowledge mechanism DK which manages the agent's knowledge is shown in Fig. 13, which consists of a message parser, message handlers and agent scripts. The message parser classifies the messages from the CM and selects a message handler to deal with the message. A message handler is a set of procedures to deal with a message based on the knowledge written in the agent scripts provided in the ADIPS framework. The general syntax of the agent script is illustrated in Fig. 14, and the detailed forms of scripts are defined according to each domain of knowledge.

The ADIPS reopsitory and the ADIPS workspace are implemented using the language C++ and Tcl/Tk in the UNIX workstations on the TCP/IP communication protocol.

4.2 Experimental System of Adaptive Distributed System

An experimental system of an adaptive video-conferencing system has been prototyped using the ADIPS framework to confirm that the adaption functions designed in the chapter 3 can be realized.

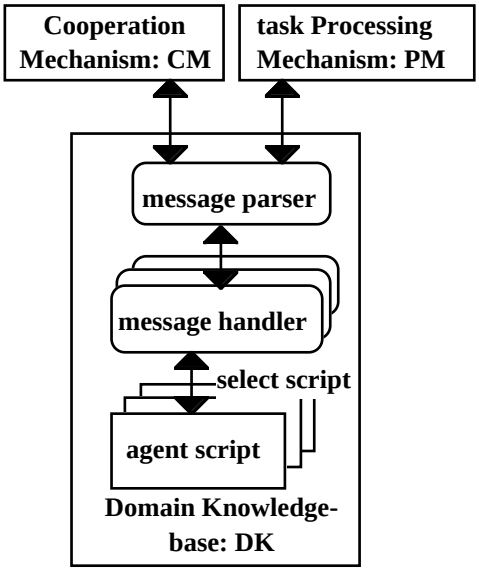

Fig.13 Domain Knowledge-base: DK

Fig.15 shows an agent-based architecture of the adaptive video-conferencing system.

A component repository CR is developed using an ADIPS repository of Fig.10, which manages the class agents as components of video-conference systems. Accoding to a request of User-A, an adaptive video-conferencing system AVCS(A) for User-A is

```
      <script> :=  "(" <script-name><id><body>+ ")" .
<script-name> :=  "Script:" name-of-knowledge .
         <id> :=  "id:" alphanumeric .
       <body> :=  "(" <condition><message> ")"
  <condition> :=  "(" "condition:" frame ")" .
    <message> :=  "(" "message:" message-description ")" .
```

Fig. 14 ADIPS script definition

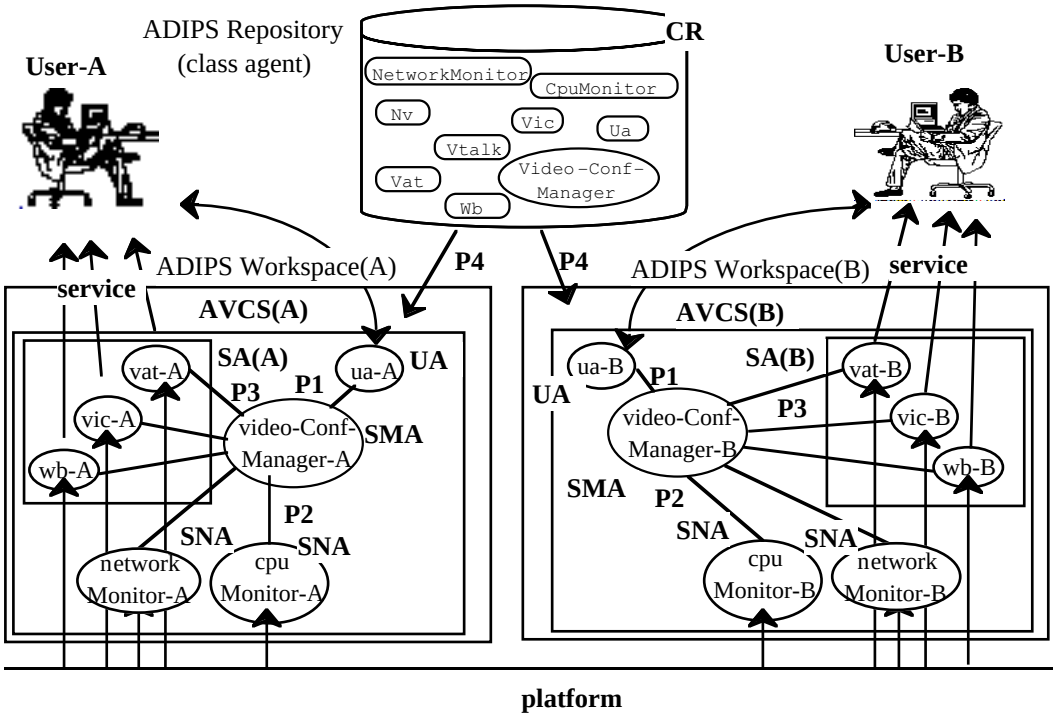

Fig.15 Experimental System of Adaptive Video Conference System

generated in the ADIPS workspace(A) installed in a UNIX workstation. An AVCS(B) is also generated as the ADIPS workspace(B). The QoS of the AVCS(A) is aquired by a user agent ua-A (UA) from User-A and sent to a video-conferencing manager agent video-Conf-Manager-A (AMA) using the protocol P1. The video-Conf-Manager-A sends an initialization message the adaptive video-conferencing service agent SA(A) based on the QoS commited with User-A. The SA(A) consists of the video service agent vic[10]-A , and the audio service agent vat[7]-A and the white board agent wb[8]-A.

The details of behavior of the protocype system are demonstrated in the next chapter.

5. Evaluation of Adaptation functions of ADS

Using a prototype of of the adaptive distributed system, i.e., an adaptive video-conferencing system, we have done the experiment on the adaptation capabilities of the prototype and we confirmed that the adaptation functions can make an agent-based video-conferencing system adaptive with respect to the changes of the operational conditions of the syste's environment. The results of the experiment are demonstrated and evaluated in this chapter.

5.1 Adaptation by Parameter Tuning Function

According to a user's request (User-A) to start a video-conference session with other user (User-B), the service modules for video-conferencing are instantiated and activated in the ADIPSworkspaces located on the respective users' workstations as shown in Fig.15. At this time, the User-A and User-B want to get the smoothness of moving picture images, hence, an agent with the Vic process is selected and instantiated as the vic-A and vic-B agents. The control parameters of the Vic processes of these agents are set to provide the color video service with the high frame rate. During the video-conference session, another application process had been started in the workstation of User-A and the slight reduction of the computational resources is detected and reported by a cpu monitor agent cpuMonitor-A based on the protocol P2. Due to the reduction of the QoS of the platform, the quality of the video service of User-A

is degraded. In this case, the video-Conf-Manager-A which has a parameter tuning function of Vic process, made a plan to realize a monochrome video service with the high frame rate. And then, the video-Conf-Manager-A sent a message to vic-A based on the protocol P3 to modify the control parameters of its Vic process. The above tuning operation has finished within about 20 minutes. As a result, the video-conferenceing service is maintained against the slight change of the QoS of the platform.

```
(Script DECISION
  id: high-traffic-01
  (condition: (Report
                from: networkMonitor
                content: (traffic-report
                  (local-area-network
                    (> collision 30))
                  (wide-area-network
                    (> packet-lost 50))
                (Requirement
                  top-priority: (video))
                (Current-SA
                  video: (vic)
                  audio: (vat)
                  whiteboard: (wb))
  (message: (Internal-Message
              script: M3
              content: (Suggestion
                         release: (wb vic)
                         create: (vtalk)))))
```

Fig.16. Example of description of ADIPS script of Video Conf-Manager

5.2 Adaptation by Component Replacement Function

Under the same setting of the video-conferencing service in section 5.1, the operational conditions of the platform had been changed for the worse and the video-Conf-Manager-A cannot display the high quality moving picture images for User-A. At the same time, the video-Conf-Manager-A receives a message from the cpuMonitor-A to inform the degradation of the QoS of the platform based on the protocol P2. According to the message, the video-Conf-Manager-A analyses the operational conditions to reduce the CPU load of User-A's workstation, and makes a plan to utileze a video coding system board instead of the software video-coding module which can be controled by the vic-A. As a result, the component replacement function of the video-Conf-Manager-A is activated based on the generated plan. Next, the video-Conf-Manager-A selects a service agent which manages the video coding system board, and creates an instance of the service agent in the ADIPS workspace of User-A based on the protocol P4. And then, the video-Conf-Manager-A releases the video-coding module and activates the service agent of the video coding system board based on the protocol P3. Exchanging the component of the video-coding module of User-A, the CPU load is reduced and the video-Conf-Manager-A can display the high quality moving picture images again. Through the cooperation between the video-Conf-Manager agent and the ADIPS repository, the video-conferencing system can adapt itself to the change of the operational conditions.

5.3 Adaptation by DS Reconstruction Function

During the video-conference session, a new request had been issued by User-A to provide a high quality video service with the high smoothness over 25 fpsf. The video-Conf-Manager-A agent recognizes that the given request cannot be processed by the parameter tuning operation because the vic-A cannot provide such a high quality service for the users. Hence, The video-Conf-Manager-A activates the component replacement function and creates a plan to find and replace the vic-A with another video service agent stored in the ADIPS repository, However, the video-Conf-Manager-A knows that the computational resources become insufficient to execute the new agent. Therefore, the DS reconstruction function of the video-Conf-Manager-A is activated, and a new plan is created to reconstruct the current video-conferencing system. In the new plan, the white board service should be abandoned and the vic-A agent shuould be replaced with another video service agent to realize and maintain the required QoS of video service.

Following to the created plan, the video-Conf-Manager-A exchanges the messages with the ADIPS repository to find a suitable video service agent by using the protocol P4. In this case, a video service agent named vtalk class agent is selected by the ADISP

repository and the instances of the vtalk class agent, i.e., vtalk-A and vtalk-B are created in the ADIPS workspaces of both User-A and User-B. Next, the vic-A and the wb-A are released, and the vtalk-A is installed, based on the protocol P3. As a result, the video-conference service module of User-A is reconstructed as shown in Fig.17. At the same time, the video-Conf-Manager-B agent also reconstructs iself using the same plan. It takes about 50 seconds to complete the above reconstruction operation. We confirm that the DS reconstruction function can make

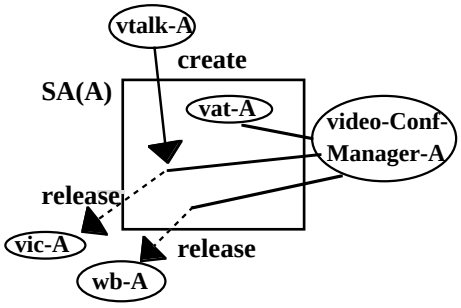

Fig. 17. Example of the re-construction

the drastic changes of the structure and functions of the system to adapt the system to the changed environment.

6. Conclusion

In order to realize the next generation distributed systems, we introduce a notion of adaptation of distributed system and propose a design model of an adaptive distributed system (ADS) in this paper. In our design model, three kinds of the adaptation methods and the adaptation functions based on thses methods are proposed. Using the adaptation functions, the ADS can provide and maintain the services for the users against various changes observed in the system's operational environment. Furthermore, we propose an agent-based architecture of the ADS based on the proposed design model and demonstrate a prototype of the agent-based ADS in the field of multimedia distributed applications, i.e., video-conferencing applications. Through the experiment of the prototype, we confirm that the adaptation functions of the agent-based ADS can make a conventional video-conferencing system adaptive with respect to the changes of both the users' requirements and the operational conditions of system's environment.

References

[1] ACM, "Special Issue: Software Agents," CACM Vol.37, No.7, 1994.

[2] Campbell, A., Coulson, G. and Hutchison, D., "A Quality of Service Architecture," ACM SIGCOM, Computer Communication Review, 24(2), pp.6-27, 1994.

[3] Feldhoffer, M., "Model for Flexible Configuration of Application-oriented Communication Services," Comp. Commun, 18(2), pp.69-78, 1995.

[4] T. Finin, J. Weber, G. Wiederhold, M. Genesereth, R. Fritzson, D. McKay, J. McGuire, R. Pelavin, S. Shapiro, C. Beck, "DRAFT Specification of the KQML Agent Communication Language plus example agent policies and architectures," http://www.cs.umbc.edu/kqml/kqmlspec/spec.html, 1993

[5] Frederick, R., Network Video, Xerox PARC Software, ftp://ftp.parc.xerox.com/ net-research.

[6] Fujita, S., Sugawara, K., Kinoshita, T. and Shiratori, N., "Agent-based Architecture of Distributed Information Processing Systems", Transactions of Informtion Processing Society of Japan, Vol. 37, No.5, pp. pp.840-852, 1996

[7]Jacobson, V.and McCanne, S.,"Visual Audio Tool," Lawrence Berkeley Laboratory, ftp://ftp.ee.lbl.gov/conferencing/vat.

[8]Jacobson, V.and McCanne, S.,"LBL whiteboard," Lawrence Berkeley Laboratory, ftp://ftp.ee.lbl.gov/conferencing/wb.

[9] Kinoshita, T., Sugawara, K., Shiratori, N., "Agent-based Framework for Developing Distributed Systems," Proc. CIKM'95 Workshop on Intelligent Information Agent, ACM-SIGART, 1995.

[10] MaCanne, S. and Jacobson, V., "Vic: a flexible framework for packet video," ACM Multimedia, pp.511-522, Nov. 1995.

[11] Magedanz T., et al, Intelligent Agents: An Emerging Technology for Next Generation Telecommunications?, Proc. INFOCOM'96, 1996.

[12] Miloucheva, I., "Quality of Service Research for Distributed Multimedia Applications," Proc. 1995 Pacific Workshop on Distributed Multimedia Systems,

pp.148-155, 1995.
[13] Nahrstedt, K. and Smith, J.M., "The QoS Broker," IEEE Multimedia, 2(1), pp.53-67, 1995.
[14] Ousterhout, J. K., "Tcl and the Tk Toolkit," Addison-Wesley, 1994
[15] Shibata, Y., Hashimoto, K. and Watanabe, M., "QoS Functions for Distributed Multimedia Network," Trans. IPSJ, 37(5), pp.731-740, 1996.
[16] Shiratori, N., Sugawara, K., Kinoshita, T., Chakraborty, G., "Flexible Networks: Basic concepts and Architecture," IEICE Trans. Comm. E77-B(11), pp.1287-1294, 1994.
[17] Shoham, Y., "Agent-oriented Programming," Artif. Intell. Vol.60, pp.51-92, 1993.
[18] Smith, R.G., "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver," IEEE Trans. Comp., C-29(12), pp.1104-1113, 1980.
[19] Sugawara, K., Suganuma, T., Chakraborty, G., Moser, M., Kinoshita, T. and Shiratori, N., "Agent-oriented Architecture for Flexible Networks," Proc. the Second International Symposium on Autonomous Decentralized Systems, pp.135-141, 1995.
[20] Turletti, T., "The INRIA videoconferencing system (IVS)," Connexions Mag., pp.20-24, 1994.
[21] Turletti, T. and Huitema, C., "Videoconferencing on the Internet," IEEE/ACM Trans. on Networking, Vol.4, No.3, pp.340-351, 1996
[22] Vogel, A., Kerherve, B., Bochmann, G. and Gecsei, J., "Distributed Multimedia and QoS: A Survey," IEEE Multimedia 2(2), pp.10-19, 1995.
[23] Wooldridge, M. and Jennings, N., (eds.). "Intelligent Agents," Lecture Note in Artif. Intell. No.890, Springer, 1995.