Videoconference System by using Dynamic Adaptive Architecture for Self-Adaptation

[1]Chulho Jung, [1]Sanghee Lee, [2]Eunseok Lee

[1,2] School of Information and Communication Engineering Sungkyunkwan University
300 Chunchun Jangahn Suwon, 400-746, KOREA
[1]{jesus98, neomine}@selab.skku.ac.kr, [2]eslee@ece.skku.ac.kr

Abstract. According to the improvements in wireless Internet technology, multimedia applications create new challenges, which must be solved. Internet-based videoconferencing systems have many variable factors, such as changes in the system operating environment or operating status of the hardware, according to the operator using the system. In this paper, we propose the Videoconferencing System, with multi-agents, for efficient videoconferencing, which is able to adapt itself to these various factors. This videoconferencing system has a dynamic architecture which is able to change the own architecture in runtime.

1 Introduction

The improvements in wireless internet technology, create new challenges which must be solved. Internet based applications are facing various factors which affect Quality of Service, the dynamically changing network environment, diverse range of user devices and preferences. Multimedia applications, especially applications in end-to-end wireless networks, need to seriously consider the dynamic network environment and system resources. Internet videoconferencing systems are no exception to this rule, and need various structures and languages for their interoperation with various network systems and operating environments, which have resulted in the development of many new technologies [1], such as Self-Adaptive software. The Self-Adaptive software [1] uses technology that is able to understand, monitor and correct changes by program itself; therefore, it should already have data relating to the program needs, the know how to enable evaluation of these data and to the ability to respond to any changes. This Self-Adaptive software technology helps clients adapt to the videoconferencing environment. For this reason, Self-Adaptive software research has been applied to many Videoconferencing Systems. [2][6]

In this paper, we propose a videoconferencing system in which the clients are able to adapt to the operator's processing capability and network bandwidth. Also, our system includes specific functions that allow it to adapt to various network situations, operating systems (O.S.'s) and devices. Moreover, the system helps a client communicates with the other clients with a different operating platform. However, it is difficult for a mobile device to adapt itself to the videoconferencing environment, as the

adaptation mechanism is optimized for a Desktop PC. Thus, we have designed dynamic adaptive architecture that changes the adaptation architecture of the devices according to the circumstances. For this, we have designed and implemented a system with an Architecture Manager, which decides and manages the client's architecture according to the particular environment.

Section 2 contains a summary of related works, in section 3 we describe the proposed system and section 4 discusses the implementation and evaluation of such software. Section 5 is a presentation of our conclusions.

2 Related Works

2.1 Dynamic Software Architecture

2.1.1 C2 – A Component- and Message-Based Architectural Style for GUI Software. [7]

C2, which was developed at UCI(University of California - Irvine), is an architecture used to support application software development, which uses a component and message based format. A C2 system is composed of a hierarchy of concurrent components interlinked by connectors – message-routing devices – such that each component within the hierarchy can only be aware of those components "above", and completely unaware of components residing at the same level or "beneath". [7] During runtime, C2 can add, delete or rearrange components, and has been optimized for flexible components.

2.1.2 Weave – Using Weaves for Software Construction and Analysis. [8]

Weaves, made by The Aerospace Corporation, are networks of concurrently executing tool fragments that communicate by passing objects, and have a dynamic, object-flow-centric architecture, designed for applications that are characterized by continuous or intermittent voluminous data flows and real-time deadlines.

Weaves embrace a set of architectural principles, known as the laws of blind communication: [8]

- No component in a network is able to recognize the sources of its input objects or the destinations of its output objects;
- No network component is able to recognize the semantics of the connectors that delivered its input objects or transmitted its output objects; and
- No network component is able to recognize the loss of a connection.

Weaves support component manipulation of a like form, emphasize the dynamic distribution, modification and rearrangement of connectors, and has been optimized for flexible connectors.

2.2 Video Conferencing System

2.2.1 JQOS – A QoS-based Internet Videoconferencing System. [2]

The JQOS system was developed at MCR Laboratory, Ottawa Univ. Canada, an is based on QoS; therefore, a QoS adaptation function is included in the Video Conferencing System

It's functions are as follows:

- First, for the End-User, the Active QoS Self-Adaptive function is now based on the network transportation capability.
- Second, for the Receiver, the smart processing of the requirements from the receiver, and the measurement for proper adaptation of the current system QoS.
- Third, for the Receiver, an expression function of the receiver's interests relating to the transporting stream is required for the QoS adaptation.

The JQOS system limits the Self-Adaptive element to stream the quality of the user, as mentioned above. This system also uses RTPR (Real-time Transport Protocol Report) as a control method, from the RTP (Real-time Transport Protocol) of JMF[4], so the Self-Adaptive function in the JQOS system is only weak.

3. The Proposed System

3.1 Architecture-based and Model-based Adaptation

Our self-adaptive videoconferencing system is an "Architecture based approach to Self-Adaptive software"[3]. In this paper, our proposed system is reflected in the above self-adaptive software process. Also, we developed the Videoconferencing system, which is operated by activity, as follows in Fig.1.

For the adaptation to a dynamic environment, all environment information, such as the system components and theirs interactive environmental information, and the available resources and operating environments, are captured and modeled with respect to several multi-aspects. The term "multi-aspect" refers to four main aspects - an architectural aspect, a behavioral aspect, a resource aspect and an environment aspect. Each aspect is defined as follow:

The architectural aspect is used to describe configurative entities, such as the software architecture, its configurative elements, each element's operation or role, and the relationship of each element. Each element is represented by a node, referred to as a "Component", with the relationship of each component represented by arcs,

which are referred to as ‘connectors’. The behavioral aspect defines the ‘Interactive’ operation end events between components.

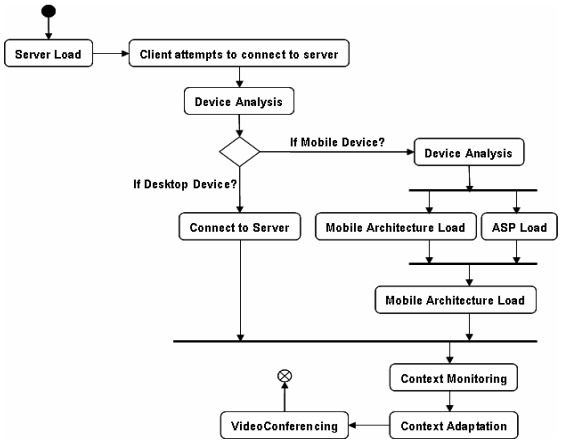

Fig. 1. Activity diagram of Our Videoconferencing System

The resource aspect defines the general system environmental entities that execute the necessary videoconferencing system. The target of a system for modeling the resource aspect is classified into two categories, the static and dynamic contexts. The static contexts are elements that have less dynamic changing characteristics, such as the type of user device, OS, executing software and Player information. The dynamic contexts are dynamic changeable elements, such as the current CPU and memory usage, state of camera and audio devices, network bandwidth and user preference. The Environment aspect defines the external environmental element that is mutually applied or affected by the surroundings of the videoconferencing system.

These four aspects of modeling are graphically and texturally represented using the GME (Generic Modeling Environment) Tool. To adapt for executing system, Modeled elements must be transmitted to be realized by software, so we applied CC/PP technology to recognize system. Fig.2 shows the CC/PP format for recognize system.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:prf="http://www.testProfile.com/profiles#ccppschema-20030226#">
      <rdf:Description rdf:ID="MyProfile">
            <prf:component>
                  <rdf:Description rdf:ID="HardwarePlatform">
                        <prf:ScreenSize>84x30</prf:ScreenSize>
                        <prf:Type>PDA</prf:Type>
                        <prf:CPU>PPC</prf:CPU>
                        <prf:Memory>16MB</prf:Memory>
                  </rdf:Description>
            </prf:component>
      </rdf:Description>
  </rdf:RDF>
```

Fig. 2. CC/PP format for recognize system.

3.2 Overall Architecture

Our general approach for supporting the Dynamic Adaptation Architecture consists of several parts: the *Server, Client, Mobile-Client and Adaptation Proxy Server*. Fig.3. show the overall architecture.

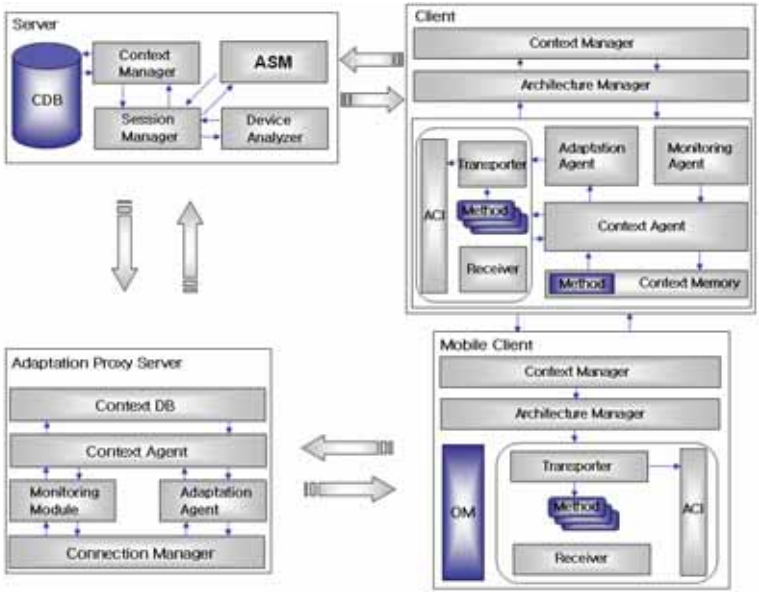

Fig. 3. Overall architecture

The detailed architecture and algorithm of each part can be described as follows:

a) Server

The server is a manager, which manages the session, Adaptive Proxy Server and context, and consists of 1) Context DB, 2) Adaptive Proxy Server Manager (APSM), 3) Session Manager, 4) Context Manager and 5) Device Analyzer. The server is loaded into Desktop Pc.

The Context DB is the stored context, by the Context Manager, of the connected client. The APSM manages the Adaptive Proxy Server. The functions of the APSM are as follows: 1) creates new APS; 2) assigns clients to APS; 3) controls client's number per APS; and 4) manages a session of executing APS. The policy of an APS creation mechanism is as follows:

1. *If there is no device with enough resource on the network, the APSM does not create an APS.*
2. *If there are devices with enough resources on the network, the APSM creates an APS.*
3. *If the number of clients per APS is larger than 15, the APSM creates a new APS. We investigate the relationships between the APS and the clients. '15' is optimized the number of clients per APS. However, if there is only one device with enough resource on the network, the APSM create another APS in this device.*

The *Session Manager* manages a client's sessions, sending session information to the others clients and requests allocation of an APS to the APSM. The *Context Manager* manages the context data in the Context DB, and the *Device Analyzer* analyzes the connecting client's device. The functions of the Device Analyzer are as follows: 1) distinguish between client's devices, 2) when the connecting client's device is mobile, the Device Analyzer requests allocation of an APS to the APSM through the Session Manager.

b) Client

The Client is a default user part, the architecture of which is organized by the Architecture Manager. The Client is loaded into Desktop Pc. When the client's resource is sufficient, the architecture of Client is structured. In cases where the resource is insufficient, the Adaptive Proxy Server and Mobile Client are used. The main function of the client is videoconferencing. The Client consists of: 1) the Connection Manager, 2) Architecture Manager, 3) Monitoring Agent, 4) Adaptation Agent, 5) Context Agent, 6) Context Memory and 7) Communication Agent.

The Connection Manager is a component that manages the client's connection to the server and sends session information to the server. The Architecture Manager is a component that manages the client's architecture, which is composes of components according to the contexts. If its resources are insufficient for contents adaptation, the Architecture Manager organizes its architecture as that of a Mobile Client's architecture, at which point the Architecture Manager uses a runtime reconfiguration mechanism. Runtime reconfiguration can be performed by altering the connector bindings, as these mediate all component communication. The runtime reconfiguration mechanism has been proposed in many researches [9, 10].

The *Monitoring Agent* gathers the client context. The *Adaptation Agent* decides the method of communication and performs contents adaptation according to information in the Context Memory of other clients. The functions of the Adaptation Agent are as follows: 1) receive the context information of each client from the Monitoring Agent, and 2) determines which communication method is used by the client according to the context information.

The decision of the Adaptation Agent is based on the following policy:

[Monitoring Agent must have only one Parameter each step. (D)(W)(PO)(A) Parameter is Available, (D)(P)(PX)(A) Parameter is disabled.]

Decision Policy

```
If the first Parameter is (P), an application module
for PDA is required. The Default Parameter is (D)

If the second Parameter is not (W), but either (U), (L)
or (S), the Direct Audio Renderer and Capturer must
change. The Default Parameter is (W).

If the second Parameter is (C) or (M), an algorithm
change is needed. The Default Parameter is (M).

If the second Parameter is (S), the environmental con-
figuration of the client for this Parameter is (W),
change is needed.
```

```
If the third Parameter is (PX), a port change algorithm
is executed. The Default Parameter is (PO).

If the forth Parameter is (NA), an algorithm for syn-
chronizing the vector number with the connect number is
needed. The Default Parameter is (A).
```

The *Context Agent* stores the contexts to the Context Memory and obtains the contexts in the other client's Context Memory through the ACI (Agent Communication Interface). The *Context Memory* stores the client's context and communication methods. The *Communication Agent* communicates with the other clients, and consists of the *Transporter*, *Communication Method* and *Receiver*.

The *Transporter* obtains video and audio information from the system, and communicates this contents to the other clients using the Communication Method. *Communication Method* is a transmission code that is received from the communication target. The *Receiver* receives video and audio information from other clients.

c) Mobile Client

The Mobile Client is a client with an architecture for mobile users that have insufficient resources for a contents adaptation. The Mobile Client's architecture is decided by the Architecture Manager, which then organizes the Mobile Client's architecture. The Mobile Client is loaded into PDA or smart phone having video camera.

A Mobile Client has less functionality then a Client because the contents adaptation function is too heavy for a Mobile Client. Therefore, to support a Mobile Client's contents, adaptation is performed by an Adaptation Proxy Server.

d) Adaptation Proxy Server

The Adaptation Proxy Server is a proxy server, which performs contents adaptation in the place of a Mobile Client, which consists of 1) Context DB, 2) Context Agent, 3) Monitoring Agent, 4) Adaptation Agent and 5) Connection Manager. The APS is loaded Desktop Pc. If a Mobile Client connects to an APS through the Connection Manager, the Monitoring Agent gathers the context and session information of the connected Mobile Client, which are then stored to the Context DB. When a client is connecting to an APS, the Context Agent updates the Context DB in the Server. The Context Agent also obtains each client's Communication Methods using the session information of the Server. The Context Agent also sends the session information, context and Communication Method to the Adaptation Agent. The Adaptation Agent decides the appropriate Communication Method for contents adapted communication. The Adaptation Agent then composes the Communication Method for the Communication Agent of the Mobile Client and the Mobile Client then communicates with each client using this Communication Method.

In this paper, we have designed a reconfigurable architecture for a Mobile Client using the dynamic link in JAVA and Weaves Style. Weaves, made by The Aerospace Corporation, are networks of concurrently executing tool fragments that communicate by passing objects, and have a dynamic, object-flow-centric architecture designed for applications that are characterized by continuous or intermittent voluminous data floes and real-time deadlines. We have already addressed Weaves [8] in section 2. Fig.4 shows the concept of dynamic component composition.

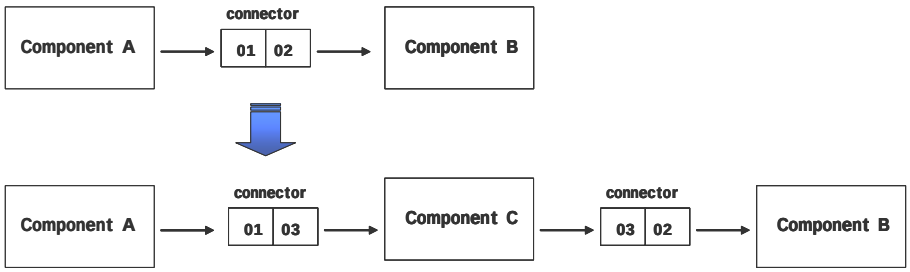

Fig. 4. The concept of Dynamic Component Composition

4. Implementation and Evaluation

The system proposed in this paper has been implemented based on JAVA SDK in Windows, Linux and Solaris environments. The main code for transmission of the picture information is made using JMF [4], for which the user needs a capturing device to obtain the picture information.

Fig. 5. Screen shot of our system

This system is implemented on a component basis, consisting of adjustable components, where each component is associated with an appropriate algorithm for adaptation to the user's operating environment as well as that of the overall network. This is the basic characteristic of self-adaptive software, which has been implemented using the Fractal [5] library for our proposed system.

Evaluation: According to the location of adaptation, an adaptation system is separated into the client-side, server-side and proxy server-side adaptation systems. [11] First, the client side adaptation system monitors and adapts the environment on its own side [12, 13], which could prevent the exposure of personnel information to the outside, and could also be suitable for performing reconfiguration of a control device configuration or to modify its functionality. Second, the server-side adaptive system monitors each client and performs contents adaptation, involving adaptive modules on their side. [14] Third, the proxy adaptation system has adaptive modules installed, which sets the proxy server between a client and server [15, 16], which can take advantage of being able to adapt, by the addition of the proxy server, without modification to the server or client components. In this paper, the difference between a general videoconferencing system and our proposed system, using the self-adaptive concept,

has been evaluated, and the quantitative differences between the use of three adaptation mechanism - Server side, Client side and Proxy Server side adaptation - based approaches and a Dynamic adaptive architecture based approach. In Fig.11, the horizontal axis represents the adaptation time, and the vertical axis the number of clients.

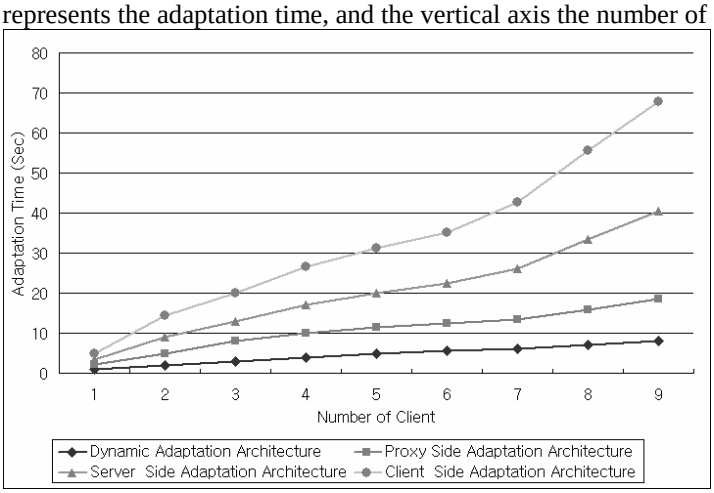

Fig. 6. Comparison of the adaptation and dynamic adaptation architectures when a mobile client is connecting. The use of dynamic adaptation architecture is more effective than those of the three adaptation mechanisms – Server side, Client side and Proxy Server side adaptation.

In Fig.6, the use of the three adaptation mechanisms required more time than that with the Dynamic Adaptive Architecture. Evaluation environment is following.

Server : HP net Server. Os. Windows 2003 Server CPU 3.0G, RAM 1024Mb
Proxy Server : Os. Windows XP Pro, CPU 3.0G G RAM 1024Mb
Client : Os. Windows XP Pro, CPU. 2.4G RAM 512Mb

An almost mobile device has less resources for adaptation; therefore, a mobile device requires a proxy. However, if mobile devices have sufficient resources, they work more efficiently than if the client performs the adaptation themselves. In view of these facts, the use of Dynamic adaptive architecture is more effective. If the number of clients per APS is optimized, our system will spend less time on adaptation.

5. Conclusion and Future Works

In this paper, architecture based self-adaptive videoconferencing system, with dynamic adaptive architecture, has been proposed for internet based videoconferencing. Our system was able to adapt to the user's environment, requests and network environment. The users of our videoconferencing system are able to utilize a system that is optimized for their environment and network surroundings. Also, this system can apply as a self-configuring and self-healing system, because it is able to cope with

new problems by the addition of new components. It also includes monitoring techniques that can deal with problems while the program is running.

In future work, this architecture will be applied to various multimedia systems and additional factors, which affect QoS, will be considered. In addition, the reconfiguration of software architecture will be focused on, and applied to adaptive multimedia systems.

References

1. R. Laddaga.: Active Software. Lecture Notes in Computer Science, Vol. 1936. Springer-Verlag, Oxford UK(2000)11–26.
2. Wenbiao Zhu, etc. al..: JQOS: a QoS-based Internet videoconferencing system using the Java media framework (JMF). IEEE ECE 2001. Vol. 1, 13-16. Canada (2001)625 - 630
3. Peyman Oreizy, etc. al..: An Architecture-Based Approach to Self-Adaptive Software. IEEE Educational Activities Department, Vol. 14 (1999) 54 - 62
4. Java Media Framework. http://java.sun.com/products/java-media/jmf/index.jsp
5. Pierre-Charles David etc. al.. : Towards a Framework for Self-Adaptive Component-Based Applications. Lecture Notes in Computer Science, Vol. 1936. Springer-Verlag (2003)1-14
6. Sung Doke Lee, etc. al..: Multi-agent based Adaptive QoS Control Mechanism in Flexible videoconferencing System. Advanced Communication Technology 2004. Vol. 2 (2004)745 - 750
7. R.N. Taylor et al..: A Component- and Message-Based Architectural Style for GUI Software. IEEE Trans. Software Eng., Vol. 22, No. 6, (1996)390–406
8. Michael M. Gorlick and Rami R. Razouk. The Aerospace Corporation..: Using Weaves for Software Construction and Analysis
9. Peyman. et al..: Architecture-Based Runtime Software Evolution. ICSE'98 (1998)
10. Quianxian Wang, etc al..: Runtime Software Architecture Based Software Online Evolution. IEEE CIMPSAC'03 (2003)
11. M. Margaritidis, G.C. Polyzos.: Adaptation Techniques for Ubiquitous Internet Multimedia. Wireless Communications and Mobile Computing, Vol.1, No.2, (2001)141-163
12. Brian Noble.: System Support for Mobile, Adaptive Applications. IEEE Personal Communications, Vol.7, No.1 (2000)44-49
13. C.Y. Hsu, A. Ortega, M. Khansari.: Rate control for robust video transmission over burst-error wireless channels. IEEE Journal on. Selected Areas in Communications, Vol.17, No.5, (1999)
14. Friday A., N. Davies, G. Blair and K. Cheverst.: Developing Adaptive Applications: The MOST Experience. Journal of Integrated Computer-Aided Engineering, Vol.6 Num.2, (1999)143-157
15. M. Margaritidis, G.C. Polyzos.: MobiWeb: Enabling Adaptive Continuous Media Applications over 3G Wireless Links. IEEE Personal Communications Magazine, Vol.7, no.6, (2000)36-41
16. IBM WebSphere® Transcoding Publisher, http://www-306.ibm.com/software/pervasive/ transcoding_publisher
17. Gu, X.; Nahrstedt, K.; Messer, A.; Greenberg, I.; Milojicic, D.: Adaptive Offloading for Pervasive Computing. IEEE Pervasive Computing, Vol. 3.(2004)66 - 73