Performance Analysis of an Error Sharing Agent running on a Multimedia Collaboration Home Study System

Eung-Nam Ko
Department of Information & Communication
Cheonan University
115 Anseo-Dong, Cheonan, ChungNam, Korea, 330-704
Korea

Abstract: - A multimedia collaboration home study system offers real time multimedia distance education system environment by home PCs and network environment. It supports real or non-real type. Also, it does not restrict on space among teachers and students into cyber-space made by network. Networks are PSTN, LAN, MAN, and WAN. It has an interaction, various communication types, a question and an answer, multi-session, application sharing, high degree of efficiency, high degree of cooperative. After a conventional method detects an error, it detects and classifies an error type by polling periodically all the process. But it is not efficient method. Therefore, the main idea of proposed method is to detect an error type by polling only process with relation to session manager. This paper explains a performance analysis of an error sharing system running on multimedia collaboration home study system using the rule-based SES(System Entity Structure) and DEVS(Discrete Event System Specification) modeling and simulation techniques. In DEVS, a system has a time base, inputs, states, outputs, and functions.

Key-Words: - multimedia collaboration home study system, error type, an error sharing system , polling process, rule-based, SES, DEVS.

1 Introduction

The need for distributed multimedia systems is growing rapidly in a variety of fields including business, manufacturing, education, CAD/ CAE, medicine, weather, entertainment, etc[1]. Different methods of a distance education include online education using PC communication, broadcasting education using TV or CATV, distance video education using broadcasting equipment and leased line, and the form of multimedia a distance education of CBM(Computer Based Multimedia) is based[2-4].

When it comes to various types of collaboration in distributed multimedia applications, the use of shared object is necessary[5,6]. These components are not always guaranteed to support enough reliability and availability for the applications. It is critical to discuss how to make and keep the systems so reliable and available that even fault-tolerant applications could be computed in the systems[7].

After a conventional method detects an error, it detects and classifies an error type by polling periodically all the process. But it is not efficient method. Therefore, the main idea of proposed method is to detect an error type by polling only process with relation to session manager. This paper explains a performance analysis of an error sharing system running on multimedia collaboration home study system using the rule-based SES and DEVS modeling and simulation techniques. In DEVS, a system has a time base, inputs, states, outputs, and functions. The purpose of this paper is to compare and analyze a performance of proposed method with conventional method by using DEVS formalism for an error sharing agent running on a multimedia collaboration home study system.

The rest of this paper is organized as follows. In section 2, Discrete event modeling, DEVS(Discrete Event System Specification) formalism ,and SES(System Entity Structure) are reviewed. In section 3, we show ECA Error Classification Agent) running on a multimedia collaboration home study system. In section 4, we propose a modeling for an error sharing agent and performance analysis of proposed method with conventional method by using DEVS formalism. Finally, in section 5, we summarize our paper.

2 Related Works

The DEVS-Scheme environment is based on two formalism:discrete event-system specification(DEVS) and system entity structure(SES)[8,9]. In this section, DEVS and SES are reviewed.

2.1 Discrete Event Simulation

Differential equations employed to describe continuous systems have a long history of development whose mathematical formalization came well before the advent of the computer. In contrast, discrete event simulations were made possible by, and evolved with, the growing computational power of computers. Since the early 70's, work has been proceeding on a mathematical formalism for modeling discrete event systems. One approach, inspired by the systems theory concepts of Zadeh and Dosoer(1963), Wymore(1967), Mesarovic and Takahara(1975), and Arbib and Padulo(1974), attempted to cast both continuous and discrete event models within a common systems modeling framework. This approach was elaborated in a number of publications primarily summarized in the books(Zeigler, 1976) and (Zeigler, 1984a), and is reviewed in (Zeigler, 1984b). Systems modeling concepts were an important facet in a movement to develop a methodology under which simulation could be performed in a more principled and secure manner. The recent advent of high performance artificial intelligence software and hardware has facilitated the transfer of this simulation methodology from research to practice(Elzas et al., 1986)[10,13].

2.2 DEVS formalism

The DEVS formalism introduced by Zeigler provides a means of specifying a mathematical object called a system. The DEVS formalism is a theoretical, well grounded means of expressing hierarchical, modular discrete event models. Basically, a system has a time base, inputs, states, and outputs, and functions for determining next states and outputs given current states and inputs. In the DEVS formalism are defined by the structure[10-13].

$M = < X, S, Y, \delta int, \delta ext, \lambda, t_a >$

where X: a set of input events,
S: a set of sequential states,
Y: a set of output events,
Int:S->S: internal transition function,
ext: Q x X -> S : external transition function
λ : S -> Y: output function
t_a : time advance function.

Basic models may be coupled in the DEVS formalism to form a multi-component model which is defined by the structure[10-13].

$DN = < D, \{M_i\}, \{I_i\}, \{Z_{ij}\}, select >$

where DN: Diagraph Network,
D : a set of component names,
$\{M_i\}$: a component basic model
$\{I_i\}$: a set, the influences of I and for each j in Ii,
$\{Z_{ij}\}$: a function, the I-to-j output transition,
select: a function , the tie-breaking selector.

2.3 SES formalism

The system entity structure(SES) directs the synthesis of models from components in a model base. The SES is a knowledge representation scheme that combines decomposition, taxonomic, and coupling relationships. The SES is completely characterized by its axioms. However, the interpretation of the axioms cannot be specified and thus is open to the user. When constructing a SES, it may seem difficult to decide how to represent concepts of the real world. An entity represents a real world object that either can be independently identified or postulated as a component of a decomposition of another real world object. An aspect represents one of decomposition out of many possibility of an entity. The children of an aspect are entities representing components in a decomposition of its parent. A specialization is a node of classifying entities and is used to express alternative choices for components in the system being modeled. The children of a specialization are entities representing variants of its parent.

For example, in an SES for a computer system, the entity printer could have such specialization as: size, typeface, and interface-type. The children of interface-type might be parallel interface and serial interface. These are variants for the interface of printer. That printers also come in various sizes is represented in the specialization size[10-14]. The properties of a SES are illustrated in Figure 1. The root entity is AB. AB is shown as having a decomposition into A and B, i.e., it is a system built from two component systems. The entities of an aspect represent distinct components of a decomposition. A model can be constructed by connecting together some or all of these components. The aspects of an entity do not necessarily represent disjoint decompositions. A new aspect can be constructed by selecting from existing aspects as desired. A has a specialization, shown by two vertical lines, called A-spec entities A1, and A2. The triple vertical bars connecting B and B-dec represent a special type of decomposition called a multiple decomposition. A multiple decomposition is used to represent entities whose number in a system may vary[10].

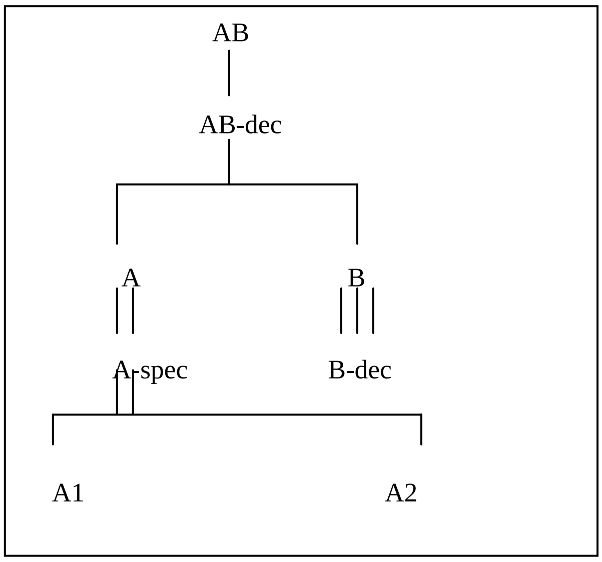

Fig. 1 The example of SES

3 An Error Sharing Agent

This paper explains a performance analysis of an error sharing system running on a multimedia collaboration home study system using the rule-based SES and DEVS modeling and simulation techniques.

3.1 a Multimedia Collaboration Home Study System

DOORAE(Distributed Object Oriented collaboRAtion Environment) is a framework technology for computer collaborative work. It has been running since April 1996 at SKKU(Sung Kyun Kwan Univ.) in Korea. It has primitive service functions. Service functions in DOORAE are implemented with object oriented concept. We call agent layer. As shown in figure 2, the organization of DOORAE includes 4 layers. The four layers consist of a communication layer, a system layer, a DOORAE agent layer, and a multimedia application layer. The communication network is being presently developed with UDP broadcasting in order to decrease communication rate and TCP/IP on the Ethernet and ATM. Additional packet form has been defined and expanded for realization of DOORAE's functions. The hardware environment of DOORAE consists of multimedia PCs, a network adapter, keyboard/mouse, image scanner, microphone, video camera, monitor, speaker, printer, video processor and accelerators. The operating system was first developed on windows 3.1 but presently windows 98, windows 2000, windows NT, and windows XP are supporting the development as well.

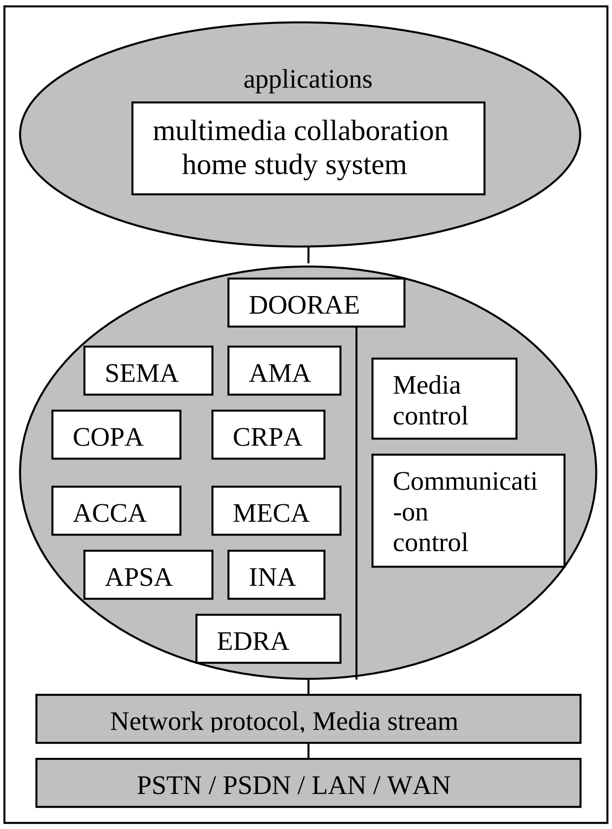

Fig. 2 The organization of DOORAE layer

DOORAE agents are composed of AMA(Application Management Agent) that handles request of application, SEMA(SEssion Management Agent) that appropriately controls and manages session and opening/closing of sessions, even in the case of several sessions being generated at the same instant, COPA(Coupling Agent) that provides participants same view, CRPA(CRoss Platform communication Agent) that manages formation control of DOORAE communication protocol, ACCA(Access and Concurrency Control Agent) that manages access control and concurrently control agent, ASPA(Application Program Sharing Agent), INA(Intelligent Agent) that manages convertible media data between IBM compatible PC and Mac, MECA(Media Control Agent) that supplies user access and convenient application.

The multimedia application layer includes general application software such as word processors, presentation tools and so on. And it supplements various functions such as video conference and voice conference for multimedia collaboration home study system

3.2 ECA

EDRA consists of EDA(Error Detection Agent), ECA(Error Classification Agent), and ERA(Error Recovery Agent). ECA consists of ES(Error Sharing) and EC(Error Classification). EDA detects an error by using hooking methods in MS-Windows API(Application Program Interface). When an error occurs, A hook is a point in the Microsoft Windows message-handling mechanism where an application can install a subroutine to monitor the message traffic in the system and process certain types of messages before they reach the target window procedure. Windows contains many different types of hook.

3.2.1 ES(Error Sharing)

As shown in Fig.3, the roles of ES(error and application program sharing) are divided into two main parts; Abstraction and sharing of view generation.

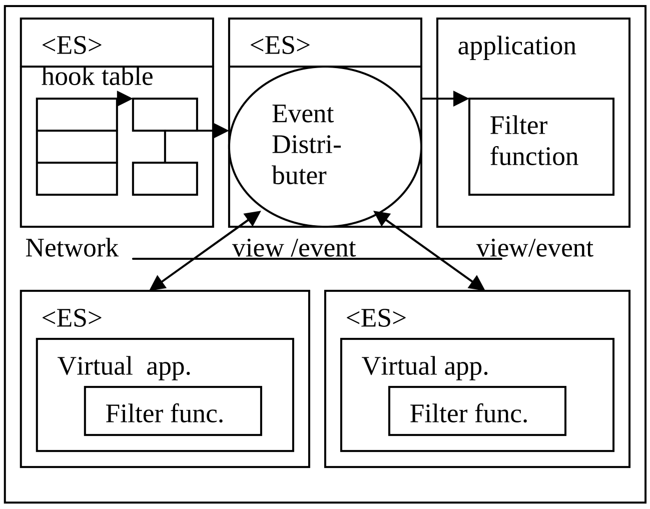

Fig.3 Error and Application Sharing Process

Error and application program sharing must take different from each other according to number of replicated application program and an event command. This proposed structure is distributed architecture but for an error and application program sharing, centralization architecture is used. An error and application program sharing windows perform process communication of message form. In the middle of this process, there are couple ways of snatching message by error and application sharing agent. ESA informs SM(Session Manager) of the results of detected errors. Also, ESA activates an error for application software automatically. It informs SM of the result again. That is, ESA becomes aware of an error occurrence after it receives requirement of UIA and transmit it.

3.2.2 EC(Error Classification)

As shown in Fig.4, you can see the organization of ECA. EDRA consists of EDA, ECA, and ERA. ECA has a function of an error classification.

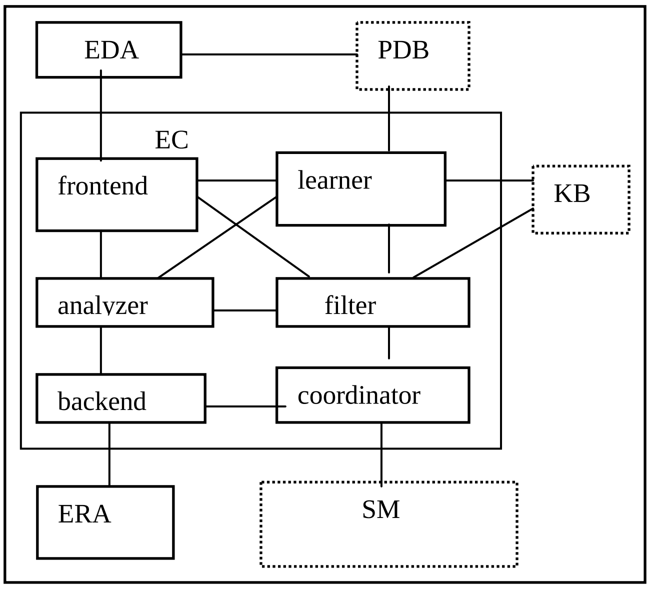

Fig.4 The organization of EC

EC consists of frontend, backend, analyzer, coordinator, filter, and learner. Frontend has a function of playing a role in receiving an error detection information from EDA. Backend has a function of playing a role in receiving an error recovery information from ERA. Coordinator informs SM of the result. Analyzer has a function of classifying error's information that is received from frontend. Learner has a function of classifying the type of errors by using learning rules with consideration of information from analyzer. Filter has a function of storing an error's history information in KB(Know ledge Base) from error information that is classified by learner.

4 Performance Analysis by DEVS

To evaluate the performance of the proposed system, an error sharing method was used to compare the performance of the proposed model against the conventional model by using DEVS formalism.

(Simulation 1)
In the first simulation, we have considered composition component as shown in Table 1. The atomic models are EF, RA1, UA1, and ED1. The combination of atomic models makes a new coupled model. First, it receives input event, i.e., polling interval. The value is an input value in RA1 and UA1 respectively. An output value is determined by the time related simulation process RA1 and UA1 respectively. The output value can be an input value in ED1. An output value is determined by the time related simulation process ED1. We can observe the result value through transducer.

Table 1 Atomic Model and State Variable

Compo nent	State Variable	Contents
EF(gen r)	Poll_int	Polling interval
RA1	Ra_re_time App_cnt1 Ra_re_t_a	Response time The number of application program Accumulated response time
UA1	Ua_re_time App_cnt2 Ua_re_t_a	Response time The number of application program Accumulated response time
ED1	Ra_re_t_a Ua_re_t_a Tat_t_a	RAaccumulated response time UAaccumulated response time RAaccumulated response time + UAaccumulated response time

(Simulation 2)
In the second simulation, we have considered composition component as shown in Table 2. The atomic models are EF, RA2, and ED2. The combination of atomic models makes a new coupled model. First, it receives input event, i.e., polling interval. The value is an input value in RA2. An output value is determined by the time related simulation process RA2. The output value can be an input value in ED2. An output value is determined by the time related simulation process ED2. We can observe the result value through transducer.

Table 2 Atomic Model and State Variable

Compone nt	State Variable	Contents
EF (genr)	Poll_int	polling interval
RA2	Ra_re_time App_cnt1 Ra_re_t_a	Response time The number of application program Accumulated response time
ED2	Ra_re_t_a Sm_t_a Tat_t_a	RA accumulated response time Accumulated time to register information in SM RAaccumulated response time + UAaccumulated response time

We can observe the following. The error type detected time interval is as follows.
Conventional method:
Poll_int*(App_cnt1 + App_cnt2)
Proposed method:
Poll_int*(App_cnt1) + Sm_t_a
Therefore, Poll_int*(App_cnt1 + App_cnt2) >
Poll_int*(App_cnt1) + Sm_t_a
That is, proposed method is more efficient than conventional method in error type detected method.

5 Conclusion

This paper explains a performance analysis of an error sharing system running on multimedia collaboration home study system using the rule-based SES and DEVS modeling and simulation techniques. The roles of ES(error and application program sharing) are divided into two main parts; Abstraction and sharing of view generation. Error and application program sharing must take different from each other according to number of replicated application program and an event command. This proposed structure is distributed architecture but for error and application program sharing, centralization architecture is used. ECA has a function of an error classification. EC consists of frontend, backend, analyzer, coordinator, filter, and learner. Frontend has a function of playing a role in receiving error detection information from EDA. Backend has a function of playing a role in receiving error recovery information from ERA. Coordinator informs SM of the result. Analyzer has a function of classifying error's information that is received from frontend. Learner has a function of classifying the type of errors by using learning rules with consideration of information from analyzer. Filter has a function of storing an error's history information in KB(Know ledge Base) from error information that is classified by learner. To evaluate the performance of the proposed system, an error sharing method was used to compare the performance of the proposed model against the conventional model by using DEVS formalism.

Our future work is to extend to autonomous agents for detecting and recovering error and to generalize it to adjust any other system.

References:

[1] Naveed U. Qazi, Miae Woo, Arif Ghafoor, A Synchronization and Communication Model for Distributed Multimedia Objects, Proceedings ACM Multimedia'93, Anaheim, California, August 1-6,1993, 147-155.

[2] Kyung E. Lim, Sung C. Ahn, and Dae J. Hwang, "Intelligent Mirroring Mechanism for Interoperability Between PC-Based Platforms", In Proc. of the IASTED International Conference on Modeling and Simulation, Pittsburgh, U.S.A., April 27-29, 1995.

[3] Bohdan O. Szuprowicz, "Multimedia Networking and Communication Computer Technology Research Corp., 1994, pp.149-175.

[4] Gil C. Park, Dae J. Hwang, "Design of a multimedia distance learning system: MIDAS", In Proc. of the IASTED International Conference on Modeling and Simulation, Pittsburgh, U.S.A., April 27-29, 1995.

[5] P. Dewan, "Tools for Implementing Multi-user Interface", Trends in Software: Issue on User Interface Software I, NewYork, John Wiley and Sons Inc., 1993, pp. 149-172.

[6] C.A.Ellis, S.J.Gibbs, and G.L.Rein, "Groupware: Some Issues and Experiences", CACM 34(I), Jan.1991, pp.38-58.

[7] Hiroaki Higaki, Kenji Shima, Takaynki Tachikawa, Makoto Takizawa, Checkpoint and Rollback in Asynchronous Distributed Systems, IEEE Infocom'97 Proceedings Volume 3, April 7-12, 1997, 1000-1007.

[8] Bernard P.Zeigler, Tae H. Cho, and Jerzy W. Rozenblit, A Knowledge-Based Simulation Environment for Hierarchical Flexible Manufacturing, IEEE Transaction on Systems, Man, and Cybernetics-Part A: System and Humans, Vol. 26, No. 1, January 1996, 81-90.

[9] Tae H. Cho, Bernard P.Zeigler, Simulation of Intelligent Hierarchical Flexible Manufacturing: Batch Job Routing in Operation Overlapping, , IEEE Transaction on Systems, Man, and Cybernetics-Part A: System and Humans, Vol. 27, No. 1, January 1997, 116-126.

[10]Bernard P.Zeigler, Object-Oriented Simulation with hierarchical, Modular Models, Academic Press,1990.

[11]Bernard P.Zeigler, Multifacetted Modeling and Discrete Event Simulation, Orlando, FL: Academic, 1984.

[12]Bernard P.Zeigler, Theory of Modeling and Simulation, John Wiley, NY, USA, 1976, reissued by Krieger, Malabar, FL, USA, 1985.

[13]A.I. Conception and B.P. Zeigler, The DEVS formalism: Hierarchical model development, IEEE Trans. Software Eng., vol. 14, no.2, 1988, 228-241.

[14]Chi, S.D., Lee, J.S., Lee, J.K. and Whang.J.H.,NETE: Campus Network Design Tool, in Proc. IASTED International Conference, July, 1997.