Web-based Application Service Management System for Fault Monitoring

Sang-Cheol Min, Tai-Myoung Chung, Hyoung-Woo Park, Kyung-Ha Lee, and Kee-Hong Pang

Abstract

Network technology has been developed for very high-speed networking and multimedia data whose characteristics are the continuous and bursty transmission as well as a large amount of data. With this trend users wish to view the information about the application services as well as network devices and system hardware. However, it is rarely available for the users the information of performance or faults of the application services. Most of information is limited to the information related network devices or system hardware. Furthermore, users expect the best services without knowing the service environments in the network and there is no good way of delivering the service related problems and fault information of application services in a high speed network yet. In this paper we present a web-based application management system that we have developed for the past year. It includes a method to build an agent system that uses an existing network management standards, SNMP MIB and SNMP protocols. The user interface of the system is also developed to support visualization effects with web-based Java interface which offers a convenient way not only to access management information but also to control networked applications.

Ⅰ. Introduction

Network technology has been developed for very high-speed networking and multimedia data whose characteristics are the continuous and bursty transmission as well as a large amount of data[1]. With this trend, users expect the best services without knowing the network environments. However, we do not have a good way of delivering the service related problems and fault information of the application services due to excessive traffic, the faults caused by channel loss, poor session management, and/or the abnormal behavior of application services in a high speed network[1, 2]. These problems set off the research activities on application service management in the network and standardization activities have become active in order to support the user with transparency and the best services.

In this paper, our paper defines the faults of application services and describes a prototyped work of web-based application service management system, called ASFDMS(Application Service Fault Detection Management System). ASFDMS described in this paper served two purposes: 1) to confirm the ability for detecting application service faults for better services, and 2) to use web-based user interfaces to simplify the complex system management problems and improve the usability and efficiency of service management. This paper is organized as following. In the rest of this section, we first define application service fault management and explains the concept of ASFDMS. Then, the ASFDMS model is described in Section Ⅱ before the details of ASFDMS design is explained in Section Ⅲ. Section Ⅳ describes ASFDMS implementation and results, and finally, conclusion and future work are drawn in Section Ⅴ.

1. Application Fault and Recovery

Application fault management includes general management functions which detect and control the abnormal status of application software in a networked environment. An application fault is classified as either hard fault or soft fault based on its criticalness. Hard fault cannot be recovered, but only restarting the system at abnormal status is the solution, while soft fault may be dynamically recovered by OS or other recovery utilities[3, 4]. Association fault of web server, for example, belongs to soft fault, because it can be recovered by re-association except restarting the software. Http port error belongs to hard fault and the web server that uses the port must be restarted.

Generally, a fault can be generated and recovered in the following three cases. One is that an application program itself could have an error and it can be resolved by modifying the application software. In this case, the application has to be corrected and recompiled. Another case is that the system with an application program could generate an error due to OS problems or hardware resource defects. It may be recovered by fixing

Manuscript received November 1, 1997; accepted November 25, 1997.

S. C. Min, T. M. Chung, K. H. Lee, and K. H. Pang were with Real-time Systems Laboratory, Dept. of Information Engineering, Sung Kyun Kwan University, Suwon, Korea.

H. W. Park was SERI(System Engineering Research Institute).

hardware or OS, then the application be restarted. The last case is that an error occurs due to communication fault or other recoverable faults which can be resolved by fault management techniques proposed in this paper[5, 6].

2. *The Concept of ASFDMS*

Networked application management systems are generally implemented in one of the following two ways. One strategy is tightly-coupling a dedicated software with the target application. In this strategy, efficiency could be easily obtained, but not flexibility. The other strategy that we are interested in is using an agent for each application to maximize flexibility. Such management system consists of a manager module and multiple agent modules that detect and report faults to the manager module. The modules are communicated with the existing network management protocol such as SNMP[9, 10]. The latter strategy has some disadvantages that it should take into account many factors at design time and fit in the standard network management protocol into the system. However, it has advantages of having good extensibility, various functionality, and high usability. The fault management system we propose consists of agents which collect the information of the application services, and ASFDMS enables users to monitor and control networked applications via well known protocol. The protocol used in this system is SNMP to poll, collect, analyze and process the information defined in SNMP MIB(Management Information Base)[10, 11]. The system also includes an integrated visualization system which enables users to understand the analyzed contents in a convenient way with the widely available web-based Java interface[12]. ASFDMS also supports various functions, such as storing and transferring real-time data, spontaneously processing event data generated by agents. Figure 1. shows the ASFDMS architecture.

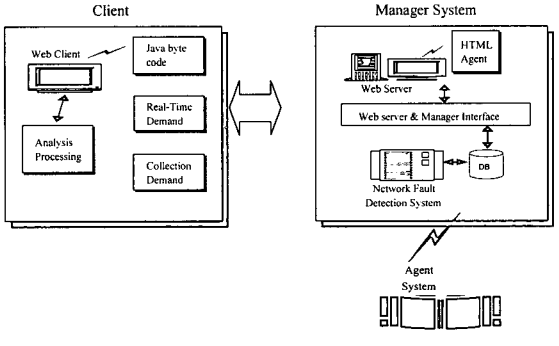


Fig. 1. Application Service Fault Management System Architecture.

3. *Related Works*

The Application MIB Working Group of IETF has defined an experimental branch of the MIB for network management, particularly objects for the monitoring and control of distributed applications in the Internet Community[7]. WebCast of NetScout Systems Inc. is a Web window announced as comprehensive suite of monitoring, reporting and troubleshooting capabilities for enterprise-wide network traffic[8].

II. A Conceptual Model of ASFDMS

ASFDMS receives requests from users, collects the information from agents, processes and stores the information into MIB database, and reports the analyzed data to users. It also controls application services through agents when a user sends a control request in the same manner that monitoring requests are processed. In this chapter, we describe a conceptual model of ASFDMS and detailed sub-model of ASFDMS. As shown in the Figure 2., ASFDMS is composed of Information Collect Module (ICM), Information Analysis Module(IAM), Integrated Visualization Controller and Database.

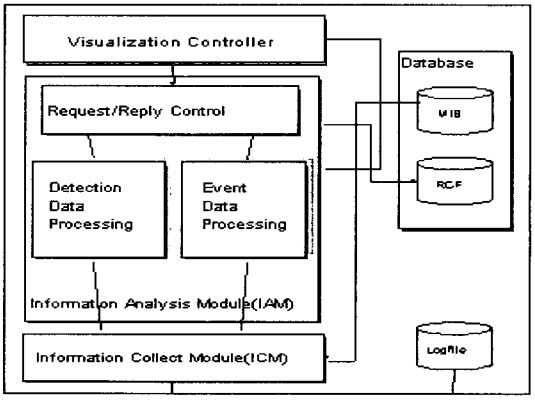


Fig. 2. Application Service Fault Detection Management System (ASFDMS) Model.

1. *Information Collect Model(ICM)*

The main roles of ICM are to receive the request transferred from information analysis module, send it to the agent, collect the information from agent, to store it in MIB, and to send the assembled contents to information analysis module(IAM). Among working modes which ICM operates, the following three cases are considered.

1) Request from users
2) Self request from ASFDMS
3) Trap information from agents

As shown in the Figure 3, ICM transforms users request into a sequence of SNMP MIB parameters and sends them to the appropriate agent in a SNMP request format. In this process, collected information is sent to ICM after storing in MIB of ASFDMS.

Self request from ASFDMS is similar to users request except that it is generated to store the reply into database without being

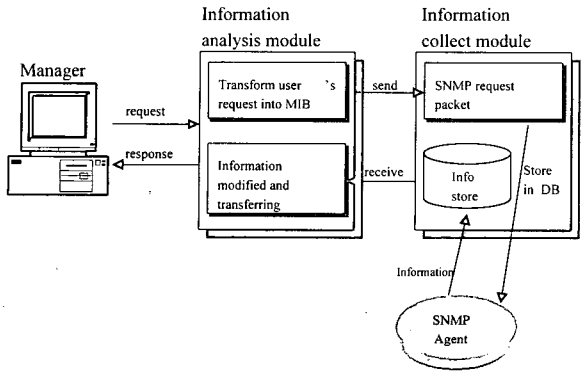


Fig. 3. Users request of information.

delivered to users. This request includes, for example, current configuration of an application service. For the third case, when a trap occurs during application service, the corresponding agent sends the information to ICM if the trap is pre-selected or in critical class. The traps are then either sent to users or handled by the ASFDMS depending on the configuration.

2. Information Analysis Module(IAM)

IAM is composed of the following three elements as shown in Figure 2.

1) *request control element* : maps users query into the corresponding MIB parameters and sends the selected MIB parameters to ICM.
2) *reply control element* : analyzes the values of parameters from ICM. The last one is *trap transmission element* that sends a trap message to users.
3) *trap transmission element* : enables manager module to provide an alarm service to users.

Besides these functions IAM has other functions such as request function for configuration update, analysis function for faults of application service, and pseudo reply function that draws raw data from logfile for reply. In particular the playback facility makes *request control element* regenerate critical fault information from logfile for non real-time analysis.

3. Integrated Visualization Controller

IVC performs interface role between users and ASFDMS. It informs users of analyzed data. In the web environment users connect to the web server that contains main page of ASFDMS, download appropriate Java applet bytecode, then perform jobs related to the application service management process such as issuing requests for monitoring or control action, analyzing collected data, or reporting results to users[12, 13]. These works are controlled by Integrated Visualization Controller.

III. The Processing Flow of Request in ASFDMS

The details processing flow of ASFDMS are shown in Figure 4 and described as the following steps.

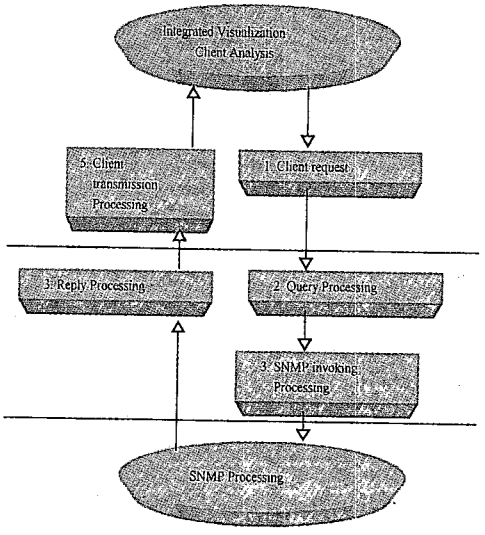


Fig. 4. Implementing Steps of ASFDMS.

1. Client Request Process

An interface between manager system and user defines exchanged data format. When a user submits a request, he(she) has to confirm to this format and then the formatted request is sent to manager system. Then, Client Request Process enables the manager system to analyze the request based on pre-determined data types and request types. The data structure that we define includes Client ID, Data type, MIB index, and Request type and generally MIB index works only when single item analysis is requested. Figure 5 shows this format type.

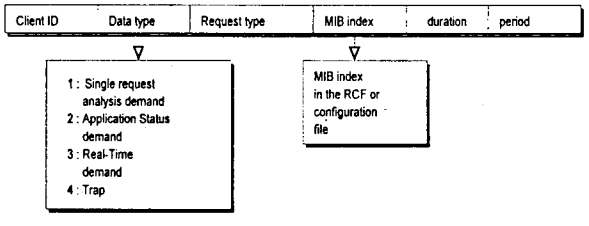


Fig. 5. Interface data format.

2. Query Conversion Process(QCP)

After analyzing the formatted request, QCP performs request conversion based information from RCF. RCF is an abbreviation of request control file which helps Query Conversion Processing to determine MIB parameters adapted to requested query and

makes new PDU confirmed to SNMP format. The functions of QCP are divided into three parts - user authentication, static/dynamic RCF file process, periodic/one-time request process. User authentication checks if a user has the right to access the system with username and password when the user sends a request. Users may request periodically or aperiodically monitoring faults of the managed application services. The request is then processed using RCF which includes both static and dynamic files[4]. Generally, users queries that demand periodic and fundamental information are converted into MIB parameters through the static RCF map table, while users queries that need many various MIB parameters use the dynamic RCF map table. The reason of having two map tables is to enhance the ASFDMSs performance by reducing workload of RCF or/and by maintaining RCF table contents into higher level memory. Periodic or one-time request is determined upon the data type specified in the request. When users request a periodic information, ASFDMS automatically takes a periodic polling with the Duration field filled, and offers the collected values to the users periodically as requested. Figure 6 depicts the RCF Process.

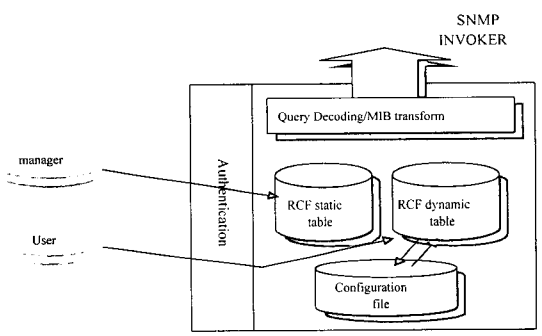


Fig. 6. RCF Process.

3. SNMP Invoking Process

After being selected by QCP, the MIB parameters are packed into a SNMP format and then transferred by the generic SNMP functions such as snmpget, snmpset. SNMP Invoker Process does this work but it doesnt use system call to invoke those SNMP functions but call snmpget, snmpset library functions in SNMP invoking process to reduce system call overhead.

4. Reply Handling Process(RHP)

RHP performs two types - MIB value processing module and trap processing module. When ASFDMS receives the values of corresponding MIB parameters from Agent, it begins reply processing. RHP converts the values of MIB parameters into the forms for users which format is above-mentioned and sends them to the users that are web clients. Converted values of MIB parameters and MIB values themselves are stored into RCF database and MIB database respectively. Trap Request Processing handles only handles the traps that are pre-defined as critical ones by system or interesting one by users.

5. Integrated Visualization Process(IVP)

Integrated Visualization Process performs the interface role between users and manager system. It informs users or managers of analyzed data. In the web environment, Integrated Visualization Process enables managers or user to connect to web server of ASFDMS, download the Java applet bytecode, then perform jobs related to the application service management process such as request data collection, analyze the collected data, and report the results to users.

IV. Implementation Environment

Implementation Environment consists of two parts, user side part and manager side part as shown in the Figure 7. User side and manager side are written in the Java language and the C language respectively[14]. Users connect to the web-server of system which include ASFDMS and get the WWW-page[12, 13] and java-applet code for accessing ASFDMS at the beginning of ASFDMS using web-server. This Java-applet code establishes a new connection between user side and manager side. It creates a new protocol which is used to communicate query. Users or managers begin to manage Application Service.

The functions of IVP are

1) establish the new association between users and ASFDMS
2) show users the results for the request with web-based user interace.

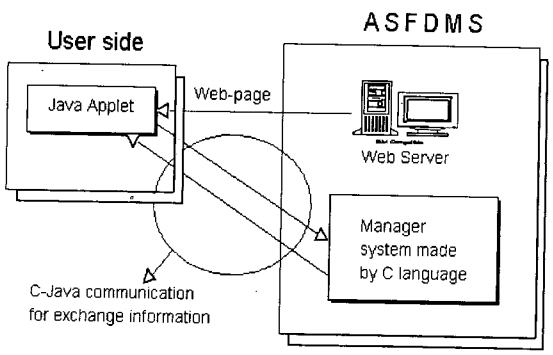


Fig. 7. Implementation Environment.

V. The Implemented Prototype of ASFDMS

Even though additional pages may be easily and quickly implemented in ASFDMS, thus far we have prototyped two initial pages for accessing ASFDMS and four basic services to show how it works. The two initial pages are *Access page* and *Initial*

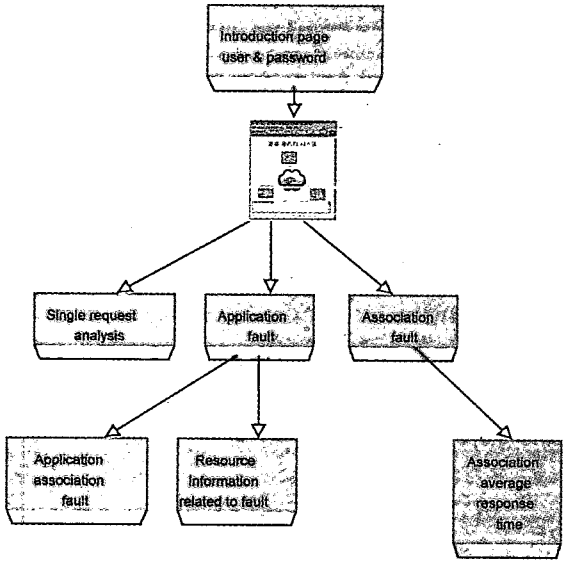


Fig. 8. Prototype of ASFDMS.

Fault page and the four basic services are Single Request Analysis, Application Fault, Association Fault, and Web fault. Application Fault service includes two subpages *application Association Fault Page* and F*ault Resource Information.* Association Fault includes a*ssociation average response time*. These implementation results enable us to consider the possibility of detecting application service faults.

1. Access Page

If a manager want to use ASFDMS, he has to connect to *Access Page* in the web, *Access Page* displays a simple introduction to ASFDMS and requests username and password for authorization. If the user has the right to access ASFDMS and supplies the correct username and password, he would be able to manage application service of fault with Initial Fault Page.

2. Initial Fault Page

ASFDMS automatically requests the status of application service to agent and basic information after *Access Page* is brought up. Then *Initial Fault Page* shows the status of application service using a graph and the basic information in the text area. The graph part of *Initial Fault Page* consists of icons for applications controlled by agent. Its outline describes the status of an application service that is one of three parts alive, dead or unknown. Blue, red and yellow outline indicates the status of alive, dead and unknown respectively. Unknown status implies that the application is configured in database but ASFDMS does not find the application service.

When the mouse clicks any application icon, the other part shows basic information of the selected application service such as service name, operation status, OS type or platform in the text

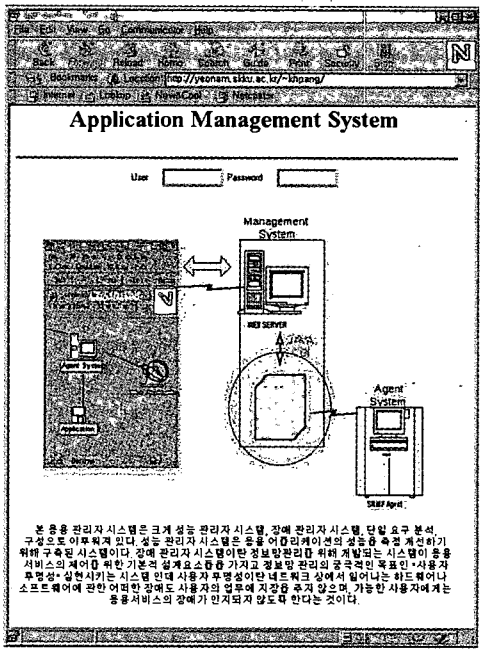


Fig. 9. Access Page

area. This page enables a manager to acknowledge the status of application service and give the alarm to the clients. Clients accept the alarm and dont request the service to the application with faults. Figure 10 shows Initial Fault Page.

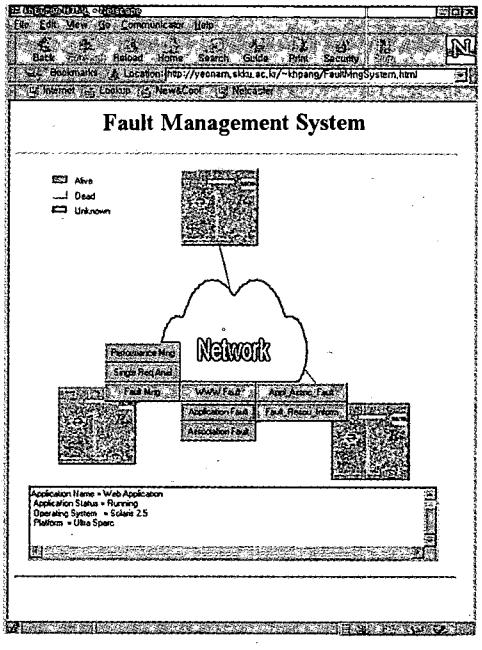


Fig. 10. Initial Fault Page.

3. Single Request Analysis

Single Request Analysis informs a manager of the raw data of

MIB. A manager uses Single Request Analysis to manage the faults of application service, which is the basis of all the management services. This facility can be exploited to the clients who want to compose the raw data to draw more sophisticated conclusion for more intelligent management service without facilities supported by ASFDMS. But manager has to get sufficient knowledge about MIB to decide the correct faults and Each manager has a different decision in the manager points respectively. The text display window displays the MIB parameter and corresponding value when one of the item blocks is selected by a user.

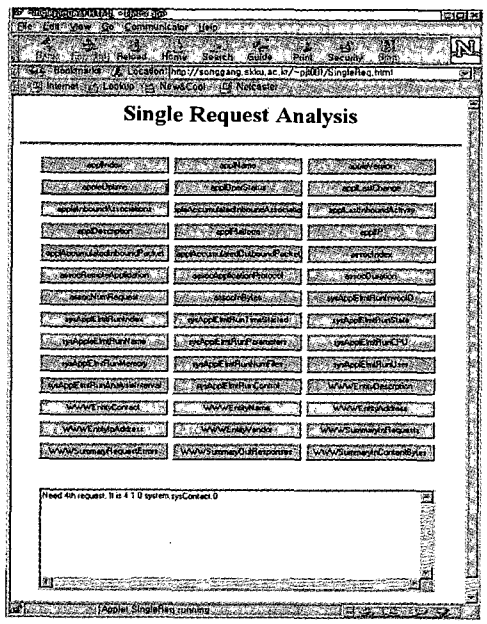


Fig. 11. Single Item Analysis.

4. Application Association Fault

Application Fault service is used to monitor the errors that occurs when the application has an insufficient resource. Among issues of application services using network, associations between clients and application services are very important. If association is unstable and insufficient network buffer size, the application using the association may get an error because it uses network to communicate for the service. Whenever www-clients accesses a new web-page, for example, they have to request a new association. Excessive requests of association increase network overload and association-reject number. As a result, these problems provide incorrect service to many clients. Therefore, manager uses Application Association Fault to detect increased association-reject number and alarm to the clients. Application Association Fault represents the status of application association and indicates users the possibility of faults using Bar graph which constitutes the associated number of clients, the replied number from application, and the rejected number of replies.

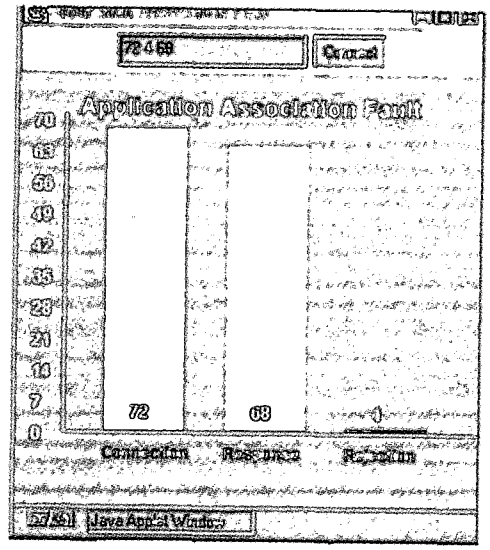


Fig. 12. Application Association Fault.

5. Fault Resource Information

In Fault Resource Information, this facility includes current queue length, average bandwidth, and average delay time of the managed application. It belongs to Application Fault because it report the information of resource related with application fault. While sufficient Queue length performs synchronization, data storage and delay jitter due to network instability, short length of queue and little bandwidth decrease application performance and delay response time. This is the reason of soft fault and user may get the fault of application. Bandwidth and the Queue can be controlled and appropriate control may reduce the application average delay time and increase the performance of application service association.

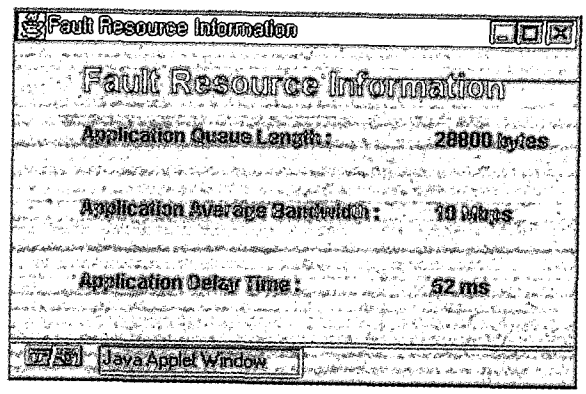


Fig. 13. Fault Resource Information.

6. Association Average Response Time

Association Fault service is for the errors that could occur when application tries to connect or to use an established connection. Association Average Response Time(AART) represents an

average end-to-end delay time for periodic real-time polling when application generates associations. For this real-time data a user should provide the total number of polling and a delay datum point. MIB data related to the average delay time are gathered periodically until the number of polling equals to the total number of polling given by the user and the polled MIB parameter (ApplAvgDelayTime) values are represented by line graph. The horizontal line in the graph means threshold. The excess higher value than threshold implies that there is a possible fault. Figure 14 shows association average response time.

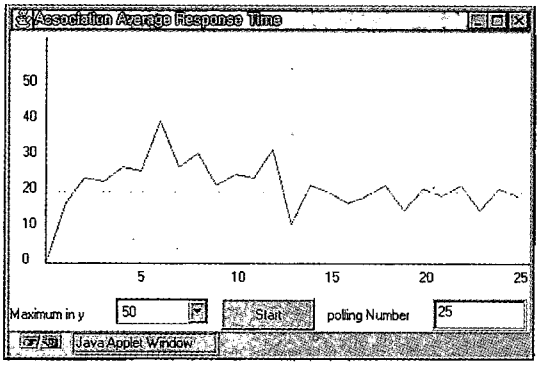


Fig. 14. Association Average Response Time.

7. WWW Fault

Web fault is caused particularly by web service. The reasons of this fault is either request or reply. The probability that faults by clients request are due to network problems and faults happened by response of application service is due to service or system fault is very high. In this points, manager acknowledges the faults and controls it. The report associated with request consists of the total number of errors detected, all the rejected number, the number of unknown message in the web, and the error rate of all requests which describes bar graph. The report associated with reply consists of the total number of responses and the total number of rejects generated by Web entity. Figure 15 shows WWW fault page.

VI. Conclusion and Future Work

In this paper, we defined the meaning of application service faults and have described the design and implementation of ASFDMS for fault monitoring. ASFDMS uses the SNMP protocol to extract fault related information of networked application services with the function of checking the status of networked application services, detecting application faults, and alarming the user. The system includes a web-based integrated visualization system that enables users to interface with ASFDMS without spending too much time learning the system. In fact, GUI is now fading out and WUI seems to be the next generation user interface. ASFDMS in the web-based integrated visualization

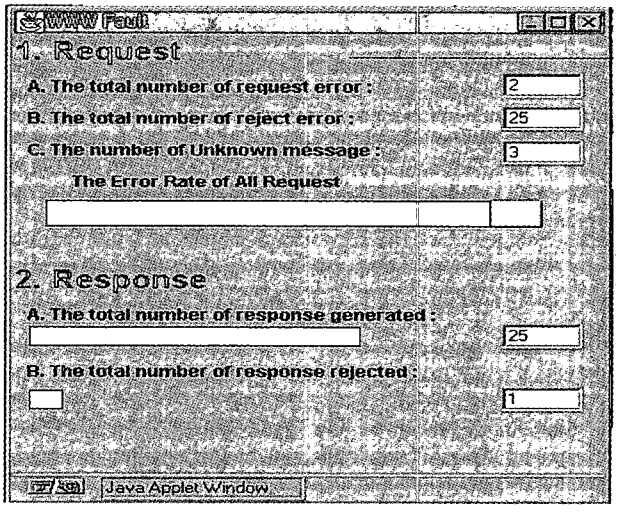


Fig. 15. WWW Fault.

system provides friendly user interface to offer end users the best services. We constructed the basic seven elements of application service faults and these elements build the foundation that provides users opportunity to get a good application service.

Even though we have attacked and solved many problems to manage networked application services, there are still more problems remaining to be solved. The following issues need to be answered or investigated for better application management service.

i) Real-time control of the traffic for faults
ii) Faults recognition and analysis of a logfile
iii) Supporting multiple agents for large scaled network environment
iv) Extended notification facility that informs users of the status of faults
v) Authentication only for authorized access

References

[1] D. Ferrari, A. Banerjea, and H. Zhang, Network Support For Multimedia-- A Discussion of the Tenet Approach, Computer Network and ISDN Systems, 26(10):1167-1180, July, 1994.

[2] R. Steinmetz and K. Nahrstedt, Multimedia Computing, Communicatioon & Applicatioon, Prentice Hall, 1995.

[3] N. Audsley, Survey : Scheduling Hard Real-Time Systems, Department of Computer Science, University of York(1990).

[4] J. A. Stankovic and K. Ramamritham, Hard Real-Time Systems, IEEE Computer Society Press, Chap1, 1988.

[5] A. Leinwand and K. F, Conroy, Network Management - A Practical perspective, Addison-Wesley, 1996.

[6] D. K. Pradhan, Fault-Tolerant Computer System Design, Prentice-Hall, 1995.

[7] C. Krupczak, Definition of System-Level Managed Objects for applicatioons, http://ds.internic.net/internet-draft/draft-ietf-

applmib-sysapplmib-08.txt, 1997.
[8] WebCast Announcement, http://www.frontier.com/press/wcast.html.
[9] J. Case, M. Fedor, M. Schoffstall, and J. Davin, A Simple Network Management Protocol(SNMP), MIT Laboratory for Computer Science, RFC 1157, May 1990.
[10] W. Stallings, SNMP, SNMPv2 and CMIP The Practical Guide to Network-Management Standards, Addison-Wesley, October 1993.
[11] M. Rose and K. McCloghrie, Structure and Identification of Management Information for TCP/IP-based Internets, Network Working Group, RFC 1155.
[12] G. Held and C. S. Horstmann, CORE JAVA, Prentice Hall, 1996.
[13] M. And C. Hughes, M. Shoffer and Winslow, JAVA Network Programming, Prentice-Hall, 1996.
[14] W. R. Stenvence, UNIX Network Programming, Prentice-Hall, 1994.
[15] T. Berners-Lee, Rfielding, and H. Frystyk, "Hypertext Transfer Protocol-HTTP/1.0", 1996.
[16] U. Black, COMPUTER NETWORKS Protocols, Standards, Interfaces Second Edition, Prentice Hall, 1994.

Appendix

MIB parameters in ASFDMS

Object ID	Object Type	Group	Description
ApplIndex	Integer	NSM	"An index to uniquely identify the network service application. This attribute is the index used for lexicographic ordering of the table."
ApplName	DisplayString	NSM	"The name the network service application chooses to be known by."
ApplOperStatus	Integer	NSM	"the operation status of the Network Application Program." 0 : running 1 : sleeping 2 : runnale 3 : idle4 : zombie: traced 6 : sxbrkstate 7 : not found
ApplInboundAssociations	Gauge	NSM	"The number of current associations to the network service application, where it is the responder."
ApplRejectedInbound Associations	Counter32	NSM	"The total number of inbound associations the application entity has rejected, since application initialization."
ApplAvgBandwidth	Integer	NSM	"An average bandwidth of the current associated clients."
ApplAvgQueueLength	Integer	NSM	"An average queue length of the current associated clients."
ApplAvgThroughput	Integer	NSM	"An average Throughput(Round Trip Time) of the current associated clients."
ApplAvgDelayTime	TimeInterval	NSM	"An average delay time of the current associated clients."
ApplAvgInPacket	Integer	NSM	"An average input packet number of the Application Program "
ApplAvgOutPacket	Integer	NSM	"An average output packet number of the Application Program"
AssocIndex	Integer	NSM	"An index to uniquely identify each association for a network service application. This attribute is the index that is used for lexicographic ordering of the table. Note that the table is also indexed by the applIndex."
AssocRemoteApplications	DisplayString	NSM	"The name of the system running remote network service application. For an IP-based application this should be either a domain name or IP address."
AssocNumRejectedRequest	Counter32	NSM	"A number of rejected request from server"
AssocBandwidth	Integer	NSM	"Bandwidth between associated client and server."
AssocThroughput	Integer	NSM	"Round trip time between the associated client and server."
AssocDelayTime	Integer	NSM	"Delay time between the associated client and server."
WWWSummaryInRequests	Counter32	WSM	"The number of requests received by this entity."
WWWSummaryRequest Errors	Counter32	WSM	"The number of requests containing errors and detected by this entity"
WWWSummaryRequest Discards	Counter32	WSM	"The number of requests discarded by this entity."

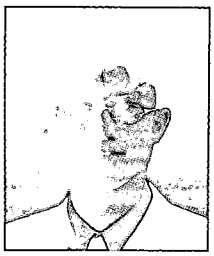
Object ID	Object Type	Group	Description
WWWSummaryResponses Discards	Counter32	WSM	"The number of responses discarded by this entity."
WWWSummaryIn UnKnowns	Counter32	WSM	"The number of unknown messages detected by this entity."
WWWSummaryInHeader Bytes	Counter32	WSM	"The number of header bytes received by this entity"
WWWSummaryInData Bytes	Counter32	WSM	"The number of data bytes receivced by this entity"
WWWSummaryOutHeaderBytes	Counter32	WSM	"The number of header bytes generated from this entity"
WWWSummaryOutData Bytes	Counter32	WSM	"The number of data bytes generated from this entity"
sysApplElmtRunInvocID	Integer	SAM	"Part of the index for this table, this value identifies the invocation of an application of which this process is a part"
sysApplElmtRunIndex	Integer	SAM	"Part of the index for this table. A unique value for each process running on the host. Wherever possible, this should be the system's native, unique identification number."
sysApplElmtRunTime Started	Integer	SAM	"The time the process was started."
sysApplElmtRunState	RunState	SAM	"The current state of the running process. The possible values are running(1), runnable(2) but waiting for a resource such as CPU, waiting(3) for an event, exiting(4), or other(5)."
sysApplElmtRunName	DisplayString	SAM	"The full path and filename of the process."
sysApplElmtRunCPU	Integer	SAM	"The number of centi-seconds of the total system's CPU resources consumed by this process. Note that on a multi-processor system, this value may increment by more than one centi-second in one centi-second of real (wall clock) time."
sysApplElmtRunMemory	Gauge32	SAM	"The total amount of real system memory measured in Kbytes currently allocated to this process."
sysApplElmtRunUser	DisplayString	SAM	"The process owner's login name (e.g. root)."

Sang-Cheol Min was born in Seoul, Korea, on August 12, 1973. He received the B.E. degree in Information Engineering from Sung Kyun Kwan University, Suwon, Korea in 1996. He is currently working on his M.E. degree at Sung Kyun Kwan University. His current interest fields are Real-Time Communication, Network Management, B-ISDN, distributed system, and network security.

Tai-Myoung Chung received his B.S. (Electronic Engineering) degree from Yonsei University, Korea in 1981, B.S. (Computer Engineering) and M.S.(Computer Engineering) degrees from University of Illinois, Chicago, U.S.A. in 1984 and 1987 respectively, and Ph.D. degree from Purdue University, Indiana, U.S.A. in 1995. Between 1985 and 1990, he worked at Waldner and Co. and at Bolt Beranek and Newman Labs. where he involved in Automated Network Management project. He is currently an assistant professor of Information Engineering department at Sung Kyun Kwan University in Korea. His research interest includes real-time systems, network management, and internet security. He is a member of IEEE and ACM.

Hyoung-Woo Park received the B.A. degree in Electronics from Seoul City University, Korea, in 1985. He received the M.S. degree in Information Engineering from Sung Kyun Kwan University, Korea in 1996. He is in the course of Ph.D. in Information Communication at Sung Kyun Kwan University and he is working for SERI as senior researcher & chiel of high performance computing networking lab. His current interesting fields are high performance computing network design & management, QoS management of Internet application, and computer network security.

Kyung-Ha Lee received the B.S. degree in Computer Science from Paichai University, Taejeon, Korea, in 1997. He is currently working on his Master's Degree at Sung Kyun Kwan University. His research interests are network management, security management and network security.



Kee-Hong Pang was born in Seoul, Korea, on September 25, 1971. Now, he is in the Department of Information Engineering from Sung Kyun Kwan University for the B.E. His research interest in network security.