

A Distributed Agent-Based Network Fault Management System for Modern Ether Local Area Networks

RUEY-KEI CHIU

Department of Information Management, Fu-Jen Catholic University

(Received Nov.6 2001; First Revised Jan.16 2002; Accepted Jul.23 2002)

ABSTRACT

Our ways of life have been pervasively influenced by the massive development of network technologies. At the dawn of the Information Age, we have seen the gradual transition from traditional face-to-face discussion and paperwork to a new way of communication via computer networking. It is now much easier and more convenient to access and share information and resources with others in this networking age. With the rapid growth of electronic commerce, electronic business, electronic banking, video conferencing, and web-based applications, the network technology has not only shortened the perceive distance of people's communications but also enhanced the efficiency of data exchange efficiency among businesses. This phenomenon heavily relies on the assistance of a reliable and robust computer network infrastructure. Consequently, how to build and manage a reliable, stable, and efficient enterprise network to effectively support business daily operation has been becoming an important and challenging issue to overcome. The network fault management may definitely be the most important issue to overcome.

FJWorks, an academic experimental research product developed by the Laboratory of Communications and Networks at the College of Management of Fu Jen University in Taiwan, is an agent-based distributed intelligent network fault management system especially designed on modern Ethernet networks. Although there exists a bunch of network management products currently being deployed in network market and research being conducted in academies and research institutes, we are trying to employ an innovative approach which combines agent-based techniques with knowledge-based system to experimentally build a more effective and efficient system dedicatedly for certain error-prone areas of network problem detection and diagnosis. Because this is only an academic research project under the restriction of available resources, our goal is not to build a complete set of network management system to compete with currently marketed products but to pay more attention on solving certain specific network fault problems, for instances a sudden lost of link connection, the abnormal reboot of a server, a duplication of IP setting on network workstations, the high network utilization during a period of time, and the occurrence of broadcasting storm. Since the functions of this research are focused on five-specific network fault areas, according to our experiment the performance is shown more appraisable than a generic network management tool. We believe to create a small and beautiful network tool may facilitate network administrators to effectively identify and solve certain error-prone network faults.

Keywords: network fault management, causal relationship, fault diagnosis, simple network management protocol, management information base

INTRODUCTION

In face of the new digital economy, enterprises have no choice but to partially or fully conduct their business operations through the network. They have thus to

sustain their survive and maintain their competitive advantages. The traditional methods of face-to-face discussion and paper-dependent communication have gradually been substituted by the way based on the Internet and enterprise networks. By taking advantages of far-reaching capability of the network, enterprises with their business partners may be more convenient to access and share their common information and resources. With the rapid growth of the Internet and the deployment of electronic business and electronic commerce, it is not only able to shorten the physical distance between people and people's communications but also able to facilitate businesses trading with the massive development of internet-based applications. All these phenomena can be effectively realized only under an environment which enterprises have a reliable and efficient internal network infrastructure that is also connected to their external environments. Consequently, how to maintain and ensure a reliable, stable, and efficient enterprise network has been becoming a substantial issue in this heavily network-rely era. Generally speaking, the issues of network management include the subjects of configuration management, fault management, performance management, security management, and accounting management [7,8]. The network fault management is believed to be the most important one of these issues to be solved. Since we are facing a rapid change of networking environment, it is very difficult for us to cope with the issues of network faults without being assisted by network fault management tools. The problems of network fault may be hundreds, but only few of them occur frequently. The current network management tools deployed in the market used for solving network fault problems are still too complicated and sophisticated to be used to effectively solve such frequently appearing fault problems under modern Ether network. We believe to have a small and beautiful network tool focusing on certain specific network fault problems may facilitate network administrators to effectively identify and solve such error-prone network faults. Therefore, the goal of this research is not to build a complete set of network management system to compete with currently marketed products but to pay more attention on solving certain specific network fault problems. Because this is only an academic research project under the restriction of available resources, the network fault problems that our research system try to solve are simply specified on five frequent appear network faults. These are the faults of a sudden

lost of link connection, the abnormal reboot of a server, a duplication of IP setting on network workstations, the high network utilization during a period of time, and the occurrence of broadcasting storm. Another goal of this research is trying to employ an innovative approach that combines agent-based techniques with knowledge-based system to experimentally build a more effective and efficient system dedicatedly for solving certain error-prone areas of network problem.

The network traffics flow through the various devices of an Ether network. They may be vulnerable to be influenced by the network faults. Therefore, to detect a network fault problem or to prevent a network from faults, first of all the traffics and their related information should be collected and analyzed in order to find the evidences that cause the fault. The traffics collected that may be caused by network faults are isolated in one network segment. A knowledge-based fault analytical model is applied to quickly identify the problem and to locate the fault area in an Ether network for a network administrator. Meanwhile, a two-phase reasoning approach is used to efficiently solve multiple problems that may occur simultaneously in a network. According to the experimental results, the approach we implemented in this research definitely shows us a better solution in conducting the network fault management from the practical perspectives.

RELATED RESEARCH AND WORKS

The computer network has been playing an important role in support of enterprise's operations. Therefore, how to effectively maintain the resources of a network (e.g., communication links, routers, switches, servers) in a normal status, utilize the bandwidth of the network, keep the network from faults, detect and recover a network once faults have occurred, etc such that the network availability and reliability can be ensured has become the major responsibility of enterprise network managers. Regardless of the circumstances, in general there are three major issues that ought to be solved once a network fault has occurred. *One* is to conduct the network fault diagnosis when there is any fault appearing in a network. *Another* is to reason the actual fault according to the traffic data collected. *The other* is to recover from fault occurrence in a network.

The occurrence of a network fault is generally viewed as an event. The event is represented into an alarm or several alarms dependent upon the event. Therefore,

once a network fault occurs, a network management system may take the alarms to identify the actual fault problem. The concept of this relationship among fault, event, and alarm can be illustrated as Figure 1. It is assumed that an Ether network is governed by TCP/IP. If a fault of broadcast storm occurs, TCP will detect the retransmission high event in the network because an unusual increase of retransmission packets is detected. TCP then gives out the alarms to SNMP monitor (simple network management protocol) to indicate heavy load in the network [5]. Under a better system design, the network management system may employ a network error-handling procedure to quickly response a problem-solving action in a form of trouble ticket so that a network specialist can take to solve the problem.

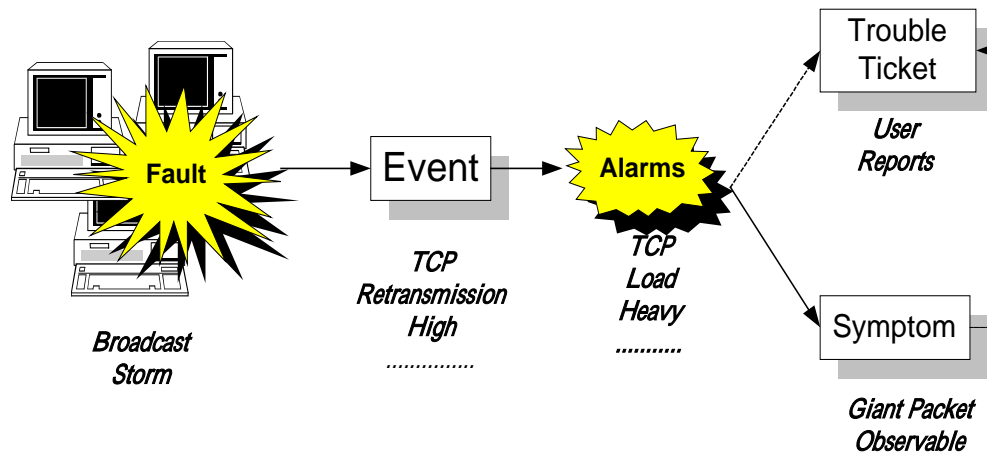


Figure 1. The Conceptual Diagram of Fault, Event, and Alarm

Maxion (1990) classified the network faults into two categories including hardware faults and software faults. The network manager's responsibility is to effectively and efficiently detect and distinguish these two faults and promptly response the errors. This can be assisted by means of an event diagnosis and alarm reasoning of a network fault management system tool. Alarms are generated by a network operating system resulting from its related events. The system also records the necessary information with regard to a possible network failure. The American National Standard for Information Technology (ANSIT) published a draft with regard to fault information characteristics in 1994 [1]. The draft defined the

procedure that an alarm is generated. The definition states that a network problem produces one or more network events, which result in alarms. The alarms may be also thought as the symptoms or evidences that partially or fully result in certain fault problems appearing on a network. Bouloutas (1994) further explained that an alarm was a measurable event that can be used to infer a specific network problem. The measurement is to indicate the probability of each alarm contributing to a specific network problem. An alarm may also be generated by more than one network fault problem depending upon the problem itself. A network management system should have the capability of inferring the fault problem based on the alarms gathered. It also needs to produce a trouble ticket for problem solving to a network manager.

Hence, Fuller (1999) divided the activities of network error control into three main tasks including *fault detection*, *fault diagnosis*, and *fault remediation*. They should be carried out in time sequence. The three subtasks are briefly described in the following sections.

1. Fault Detection

The task of fault detection is to effectively and completely detect all faults regardless that they are caused by network hardware or software failures. In addition to detect network faults, this task also needs to identify the characteristics of each network fault to which the subsequent tasks can be accurately proceeded. The characteristics of network fault are represented in messages called alarms or symptoms.

2. Fault Diagnosis

The major goal of fault diagnosis is to localize and identify network problem according to the fault characteristics identified in the task of fault detection. After collecting the current network symptoms, the network problem can be inferred by conducting an analysis that matches current fault characteristics with the historic statistical and fault patterns. The ultimate goal of this task is trying to find the real problem to cause the fault. A solution to the problem should be recommended to the network manager.

There are considerable amount of fault diagnostic models have been proposed by researchers and network management experts in the past few years. Many of them are presented based on the techniques of artificial intelligence, especially the techniques of expert systems [6]. The expert system techniques are either rule-based [3] or case-based [9]. Each of them has its special applications. Lo (1998) presented another approach to detect the network problems by using a finite state machine. The finite state machine is usually applied to detect and infer network problems when the fault information is insufficient or the problem is exceptional so that they can't be inferred in a certainty.

3. Fault Remediation

Once the network problem is detected, the network fault should be able to be effectively recovered and the network operation should return to normal operating status. The problem with its resolution identified in the execution of fault diagnosis is then given out to the network manager to solve the current network fault. The resolution may be shown in many forms. Regardless the form shown, it needs contain the problem itself, what the problem is caused by, and what are the procedures to solve the problem. The procedures to solve the problem may include several alternatives which are organized into an order for choosing from high to low priority.

THE ARCHITECTURAL MODEL OF THIS RESEARCH

FJWorks is a knowledge-based network fault management tool, especially designed for modern distributed local area networks. It was developed by the university-own laboratory of communications and networks at Fu Jen University in Taiwan. Because of the unique architectural model designed in its fault detecting engine and network-related knowledge management subsystem shown in Figure 2, its capability in network fault detection and recovery is quite effective and efficient in certain aspects compared to several other analogous systems that are currently commercialized.

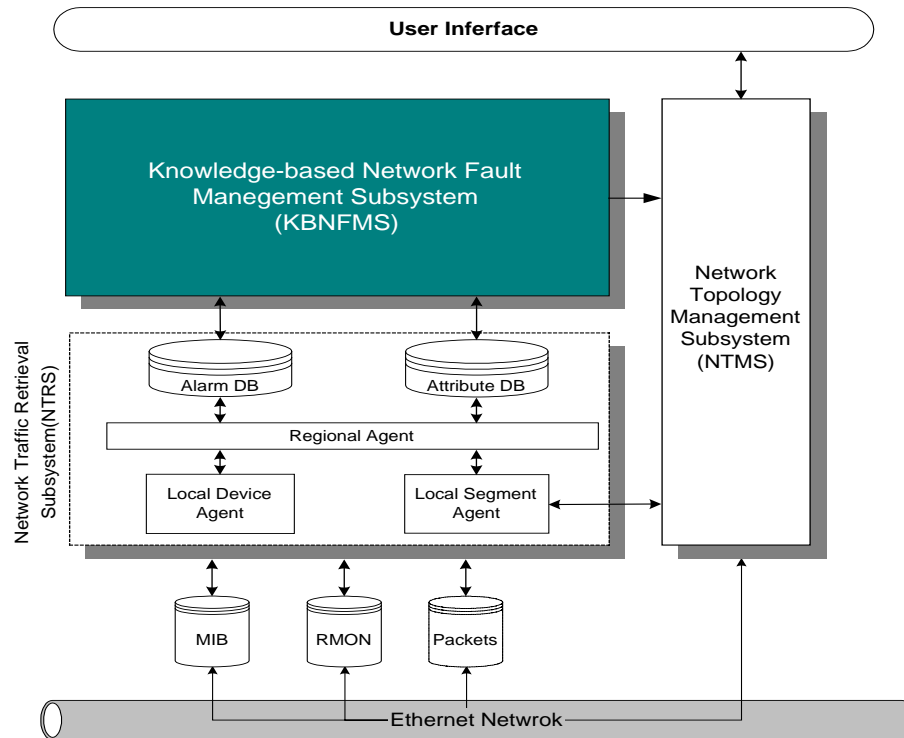


Figure 2. The Architectural model of FJWorks

As shown in Figure 2, the function of FJWorks basically consists of three main components including the Network Topology Management Subsystem (NTMS), the Network Traffic Retrieval Subsystem (NTRS), and the Knowledge-based Fault Management Subsystem (KBFMS). The major functions of these three subsystems are described in the following.

The *NTMS* is responsible for discovering the topology of the underlying network on which the system runs. It is implemented as a knowledge-inherent graphic system. Each SNMP device's historic operational status including both dynamic and static data is deposited with its corresponding node on the topology for subsequent reference and analysis.

The *NTRS* is responsible for collecting network traffic information and for identifying potential fault event. It is implemented as distributed proactive agent-based technologies. There are three types of agent including *regional agent*, *local device agent*, and *local segment agent*. These three types of agents are implemented in a distributed way and associated with each major network device and segment in which SNMP (simple network management protocol) is embedded.

The local device agent is a MIB information collecting agent responsible for acquiring network traffic-related information from the management information base (MIB) associated with each SNMP network device. **The local segment agent** is a packet capture agent responsible for catching the packets flowing through each network segment like a network traffic probe. The MIB information and segment packets caught by two local agents separately are then forwarded to **the regional agent** for potential fault event analysis and identification. Therefore, The regional agent is a network traffic analytic agent responsible for packet analysis and node symptom analysis. Once the abnormal or fault event is detected, an alarm or alarms corresponding to the event are generated and then passed to the KBFMS subsystem for further fault inference and reasoning. Basically, these three agents will communicate with each other and are seamlessly coordinated by the regional agent.

The **KBFMS** is responsible for finding out all possible faults that occur in each network device according to the alarms received from the NTRS. The network faults are then inferred according to the alarm-fault mechanism [2,5] shown in Figure 3. In Figure 3, S_i 's are alarms (or symptoms) generated by NTRS. Each P_j refers to the fault problem that may be inferred by KBFMS if its corresponding alarms are satisfied. After each network fault problem is identified, the proper actions (R_k) to solve the fault problem is then suggested. The historic data of each fault problem with its resolutions is also deposited into the topological knowledge base associated with each device node on the topology.

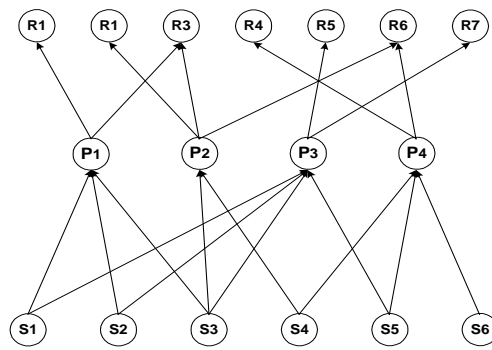
Each fault problem always consists of several alarms (or symptoms). In Figure 3 as an example, we can represent the relationship between each fault and its alarms. It is assumed that there are six alarm symptoms to raise four possible network fault problems. Their causal relationships are represented into following four set relations:

$$P1=\{S1,S2,S3\} \text{ 、 } P2=\{S3,S4\} \text{ 、 } P3=\{S1,S2,S3,S5\} \text{ 、 } P4=\{S4,S5,S6\}.$$

We may further encode these set relations into bit-oriented relation as follows:

$$P1=\{1,1,1,0,0,0\} \text{ 、 } P2=\{0,0,1,1,0,0\} \text{ 、 } P3=\{1,1,1,0,1,0\} \text{ 、 } P4=\{0,0,0,1,1,1\}.$$

This bit-oriented relation is then used for conducting the case-based fault reasoning.



Where S_i -alarm i , P_j -Problem j , and R_k -solution action for problem

Figure 3. Alarm-Fault Causal Relationship Diagram

The case-based reasoning algorithm is used in KBFMS for the fault inference because of its applicable capability of fault discovery. The case base is organized into a knowledge-based model. It also has a self-learning capability that enables it to be adopted in more complicated network environments. The main functional units of KBFMS and their correlations are illustrated as data flow diagram shown in Figure 4.

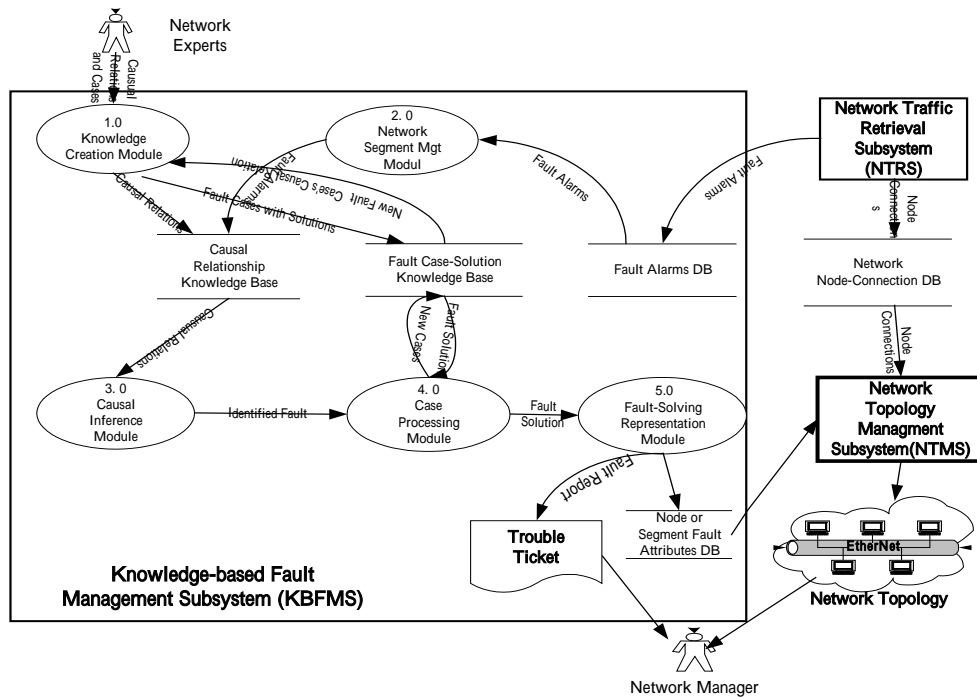


Figure 4. The Mechanism for Network Fault Reasoning and Detection

The entire network fault detecting and reasoning process, that is the process of KBFMS, consists of five major modules including the knowledge creation module, network segment management module, causal inference module, case processing module, and fault-solving representation module. The fundamental function of each module is briefly described as follows:

- 1. Knowledge Creation Module.** This module enables a network management expert to add knowledge into, delete useless knowledge from, and modify obsolete knowledge to a network management knowledge base according to different network environments. The knowledge includes network fault causal relations and well-known fault cases with solutions. The system also provides the capability of learning new knowledge that does not initially exist but is identified by causal inference module and case processing module. The entire system knowledge base is separated represented in three separate knowledge repositories. These include the *causal relationship knowledge base (KB)* and *Fault case-solution knowledge base*. Thus, the system can provide a best-fit network fault diagnostic and fault-solving model to conduct network fault resolution.
- 2. Network Segment Management Module.** This module provides the function of receiving and collecting the network fault relative alarms that are generated by NTRS. The alarms are generated by NTRS from the information recorded in the management information base (MIB) of a SNMP device and the packets, which flow through a network segment, captured and analyzed during a certain prescribed period of time. These alarms are then transferred to the causal inference module.
- 3. Causal Inference Module.** The major function of this module is responsible for inferring the network's potential faults by reasoning the alarms received from the network segment management module with the causal-relation knowledge stored in the causal-relationship knowledge base. Each network fault diagnosed is subsequently input to the case-processing module to search a best-fit fault-solving solution for each one. If the diagnosed network fault does not already exist in the causal-relationship knowledge base, it will be viewed as a new problem. The newly generated network fault with its

associating causal relation will be feed back to the knowledge creation module to store in the causal relationship knowledge base as a newly generated causal-relation knowledge.

4. Case Processing Module. This module is responsible for searching a best-fit network fault resolution by matching the network fault identified by causal inference module and its corresponding resolutions stored in the case knowledge base are retrieved if it has existed. The case-based reasoning mechanism is used to conduct this matching process. The fault-solving solution is then represented in a form of trouble ticket as shown in Figure 5 by fault-solving representation module.

The screenshot shows a web-based form titled "A Trouble Ticket" with the following sections and fields:

- Management Information:**
 - Entry-ID: [Text Box]
 - Fault ID: [Text Box]
 - Fault Date/Time: [Text Box]
 - Condition: Red, Orange, Yellow
 - IP Address: [Text Box]
 - Device Name: [Text Box]
 - Device-Type: [Text Box]
 - Submitter: [Text Box]
 - Create-date: [Text Box]
 - Notify-method: None, Notifier, E-mail
 - Assigned-Priority: Low, Medium, High
 - Assign-to: [Text Box]
 - Last-modified-by: [Text Box]
 - Modified-date: [Text Box]
- Network Diagnosis Information:**
 - Fault Name: [Text Box]
 - Fault Description: [Text Box]
 - Correlation Symptoms: [Text Box]
 - Probable Cause: [Text Box]
- Network Resolution Information:**
 - Resolution: [Text Box]
 - Ticket Status: Good, No Good, In Progress
 - Resolution Status: New, Assigned, Rejected, Closed

Figure 5. The Trouble Ticket of Fault Solution

5. Fault-Solving Representation Module. This module provides a visualized graphic representation interface in a form of trouble ticket shown in Figure 5 so that a network manager may access network information to quickly and correctly locate the network fault problem and to effectively solve it. The node or segment fault attributes are then stored in the node or segment fault

attribute database. The fault attributes to a specific node or segment are then deposited by NTMS to each corresponding node or segment in network topology. By this way, the network manager can identify each node or segment's historic fault status while browsing the network topology. It can also be used for fault predication for a long period observation of network management.

Figure 6 shows this network fault detecting and reasoning process in a more concise way of using system flow chart.

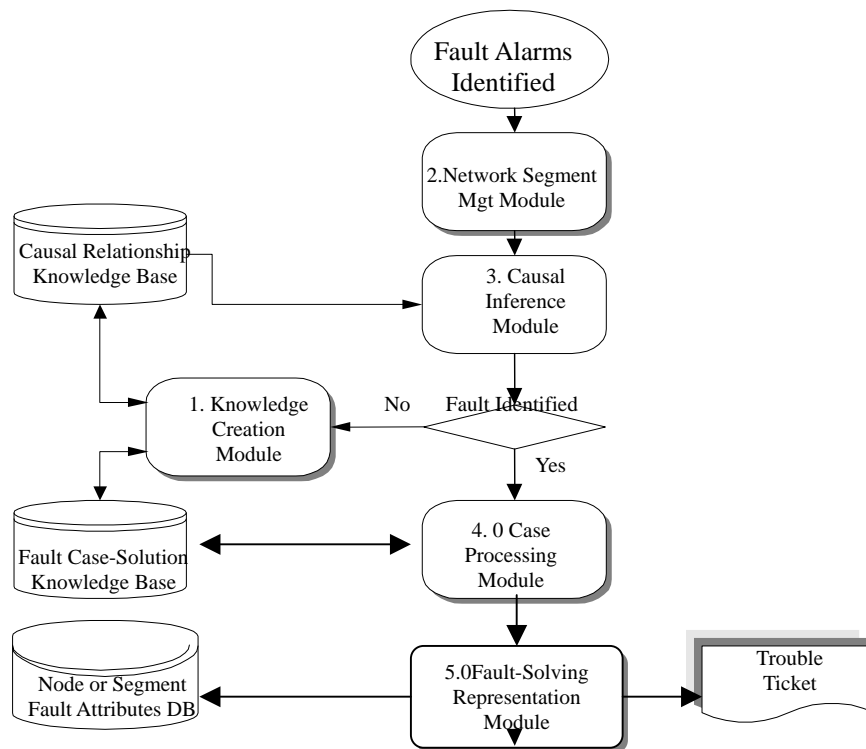


Figure 6. The System Flow Chart for Network Fault Diagnostic Process

The NTMS is responsible for discovering and managing the topology of a network. It uses a web-based graphic user interface to represent the network topology so that a network manager can effectively monitor and manage the network. The network topology discovered derives from the node-connecting message that is recorded in the routing table in each device's MIB. This approach is different from many analogous systems that use ICMP packets to identify the connectivity of the active devices in a network to analyze the network topology.

The subsystem is also implemented as an information-based system so that the static and dynamic status attributes are recorded accompanying with each network device on the topological interface. The steps of topology discovering algorithm is illustrated in Figure 7. The topology is also illustrated into a similar hierarchical diagram in Figure 8.

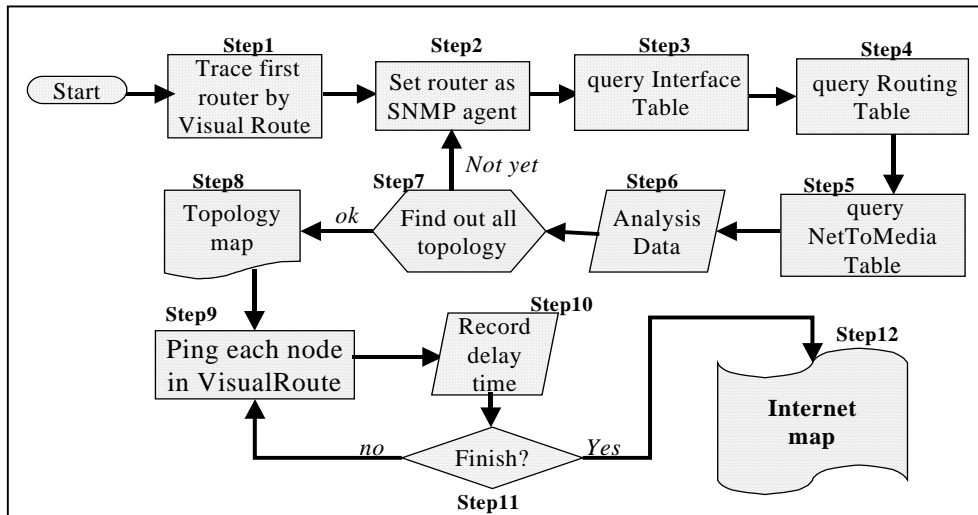


Figure 7. The Steps for Network Topology Discovering

Figure 8 illustrates a generic network topology drawn in a diagram of upside down tree. The topology is generated by the network discovery algorithm. On the map the rectangles with uppercase characters inside refer to SNMP-enabled devices, for instances, routers, switches, hubs, or servers. The notations of Sn's refer to the network segments.

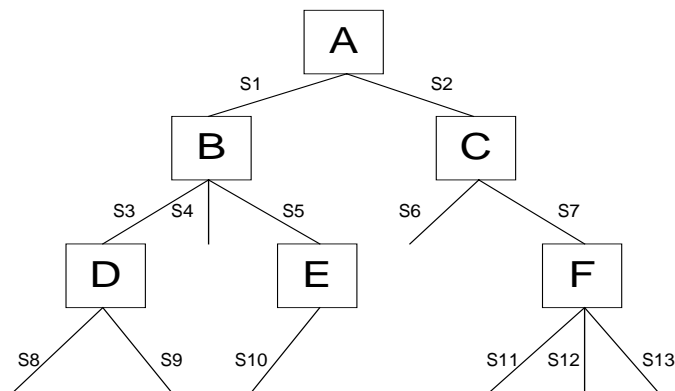


Figure 8. Generic Network Topology Map

In general, each device shown on network topology in Figure 8 is treated as a domain manager. The domain manager may be a router, a switch, a hub, or a server with SNMP embedded. The map illustrated in Figure 8 consists of six domain managers and thirteen network segments. The domain manager with its control segments may be redrawn as a domain containment like Figure 9. Each domain manager is responsible for monitoring and controlling its underneath network segments. During the network fault diagnostic and recovering stage, the faults diagnosed in the lower level domain managers are passed to their common parent domain manager for aggregation and further analysis. For example, the domain manager B will receive the diagnostic results from domain managers D and E underneath it. Therefore, domain manager A will have a global view of the fault diagnostic results of the entire network.

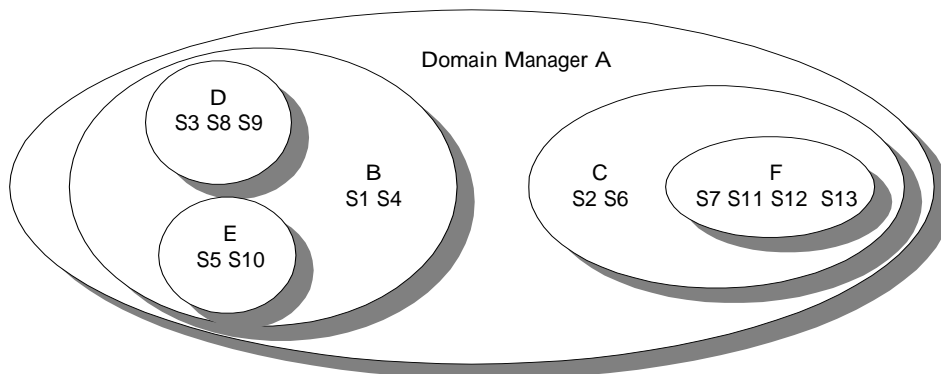


Figure 9. Network Domain Management and Views

EXPERIMENTAL RESULTS AND VERIFICATION

We implement this system in Java language and test the system on an experimental LAN that is the part of a university campus network at FJU. The experimental network structure is installed as an extended star LAN schema as shown in Figure 10. The backbone topology of this network consists of one Cisco 2501 router and one Cisco 1900 switch connected by one switchhub to extend its connectivity. Several workstations and a printer either connect to switchhub or switch. Sniffer, a well-known network management tool provided by Network

Associates Corporation, is used to generating the test traffic thus to emulate a real network environment. Five well-known network fault scenarios are experimented. These five network scenarios with their experimental results are described in below. Meanwhile, to verify the effectiveness and efficiency of solving these five well-known network fault problems, the experimental results are compared to the same experiment by using an existing network management tool, called CCWorks. The results shown in Figure 11 indicate that our system has advantages in certain aspects. Because our system's NTMS is implemented as an information-based topology, a network manager can thus conveniently access the static and dynamic information of each device of the network. Meanwhile, the network manager can keep track of the status for each network device so that the network faults can be prevented and avoided in beforehand.

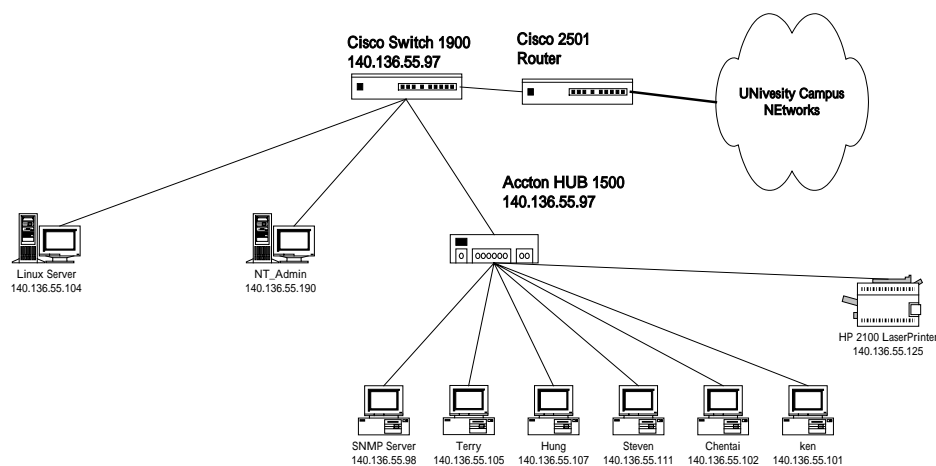


Figure 10. The Experimental Network Environment

As we mentioned in the beginning of this paper, we are not trying to develop a sophisticated network management system to solve most of the network fault problems. Our goal is to develop a specific network management tool just focusing on solving certain few but frequent appearing network fault problems such that they can be more effectively solved than using a complete and sophisticated network management tool that is currently marketed. Under each scenario experimented, we investigate each network device status and acquire the alarms

from the message recorded in its SNMP device's MIB. The packets flowing through each segment port are collected and further analysis is conducted. As we mentioned before, only five network well-known network fault scenarios are experimented. These five fault scenarios are briefly described as follows:

- 1.Sudden lost of connection (P1).** We disconnect the physical link between a workstation and its connecting network device (a switch or a router depending upon the connection).
- 2.Server Reboot (P2).** We frequently reboot a server that is being accessed by a workstation during a short time period. The status of the server is observed and investigated through its MIB.
- 3.IP duplicate (P3).** We set the IP address of a host to the same as another host on the network. We collect the network's status and observe any change that takes place in the network's operation.
- 4.Utilization high (P4).** We use the network tool Sniffer to send out overloading traffics to a network segment during an experimental period. We make the traffic high enough to degrade the network's normal performance. We then observe the network's performance after the occurrence of network traffic jam.
- 5.P5-Broadcast storm (P4).** We again use Sniffer to continuously send out broadcast packets every one thousandth of second. We make the packet traffic high enough to become a broadcast storm over the presetting traffic limit. We then observe the status of network operation.

The evidences sufficient to reason each fault problem is presented in Table 1. We experiment these five scenarios in different circumstances during a corresponding single time period. One circumstance is simulated for the occurrence of a single problem separately conducted in an experiment. The other circumstance is simulated for the occurrence of multiple problems in a single experiment. After conducting a series of experiments for each scenario under different circumstances, the experiment results are shown in Table 2.

Table 1. The Results of Experiments

Network Faults	Evidences (Alarms) for Fault	OSI Layer Belongs
P1:Sudden lost of connection	Connection lost; AllMIBNotFound; lcmplnMsgs; cmplnDesUnreaches	Layer 1
P2:Server Reboot	MIBinitiate; Connection lost; Device up	Layer 1
P3:IP duplicate	Connection lost; AllMIBNotFound; IpNetToMediaPhysAddress IpNetToMediaType	Layer 3
P4:Utilization high	Utilization high	Layer 2
P5:Broadcast storm	Broadcast storm; ifInOctets; IfInNucastPkts; ifOutOctets ifOutNucastPkts	Layer 2

According to the evidences for each fault shown in Table 1, the system's initial causal relationship knowledge base may be built by the knowledge creation module of this system. The network fault diagnostic experiment can then be conducted. The experiment is conducted in two rounds. During each round of experiment for various network fault scenarios, the alarms generated by each fault are extracted and gathered by each device local agent of NTRS from each SNMP device's MIB and are captured and interpreted by each segment agent from the packets flowing through each segment port as well. The faults alarms gathered by NTRS are subsequently employed by the causal inference module of KBFMS to reason the network fault problem. The causal relationship knowledge base provides the entire inference base during the fault reasoning. After conducting a series of experiment, we obtain a surprisingly high accurate rate of fault detection of our system. The experimental results of two-round simulations are shown in Table 2. Although the two-round experiments are conducted in different circumstances, the results show they have very consistent and identical results.

Table 2. Experimental Results

Experiment Rounds	Network Faults	# of Simulation	# of Accurate Inference	Average Accuracy
Round 1	P1	10	9	0.9
	P2	10	9	0.9
	P3	10	9	0.9
	P4	10	10	1
	P5	10	9	0.9
Round 2	P1	10	8	0.8
	P2	10	9	0.9
	P3	10	10	1
	P4	10	9	0.9
	P5	10	9	0.9

Note: P1 : sudden lost of connection , P2 : server reboot , P3 : IP duplicate , P4 : utilization high , P5 : broadcast storm.

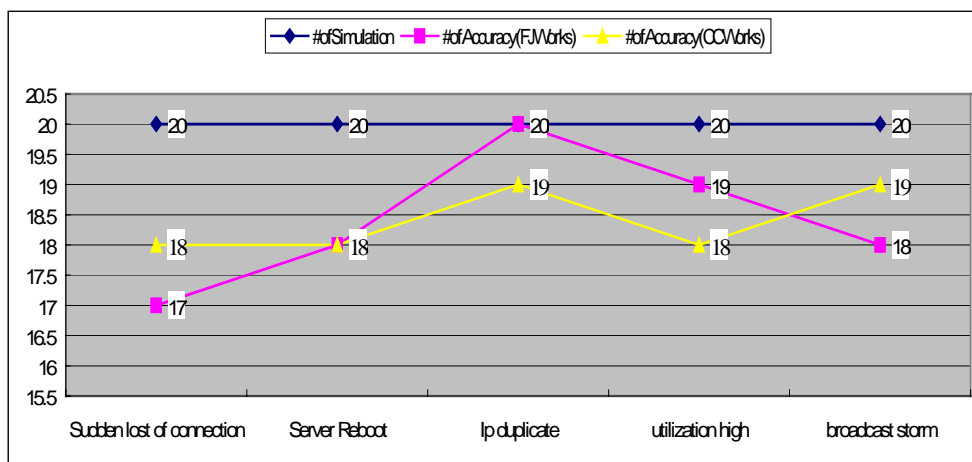


Figure 11. Comparisons of Experiment Results

The effectiveness of this system is furthermore compared to one of the most popular fault detection tool (we anonymously name it as CCWorks) currently deployed in the market. Through a series of counterpart experiments for five network fault scenarios, we found that our system provides a very competitive result. The comparing results are shown in Figure 11. The results show that our system performed better than CCWorks in two aspects of fault detection. These are

the faults of IP duplication and utilization high. It shows the same result in detecting the fault of sever reboot. Despite that the performance result shown by our system is not perfect, it is promising and encouraging.

The effectiveness of the system was examined afterwards. We found that the time needed to collect the data has a critical factor to influence the result. Two circumstances may be examined to explain this hypothesis. If the fault's appearing time period is shorter than the time interval of two adjacent packet captured (e.g., five-minute interval) as shown in Figure 12 or the fault's appearing time period crossing over two adjacent time of packet captured as Figure 13 shows, then the sufficient and necessary information may not be gathered for reasoning, thus resulting to such deficiencies. It is assumed that the periodic fault diagnostic checkpoint is scheduled in a five-minute period. In Figure 12, the broadcast storm appearing period is between within two adjacent checkpoints and it doesn't be effectively detected, thus the fault is ignored. In Figure 13, before the checkpoint at time 1:10, the connection to the server may shortly be disconnected for certain unexpected reason thus resulting in a server reboot between checkpoints 1:10 and 1:15. The fault of connection lost may be undetected at 1:10 whilst the fault of server reboot caused by the connection lost is detected at 1:15. Thus, the true fault originator can't be really identified. These missing diagnosis problems may be improved by shortening the packet captured interval. However, it also will cause the serious performance decrease to the entire network.

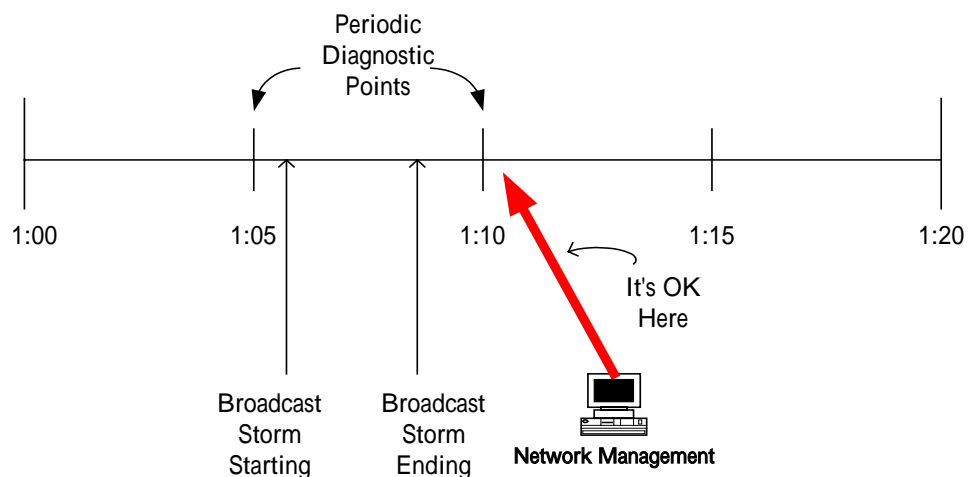


Figure 12. Fault Appearing Period Not Long Enough

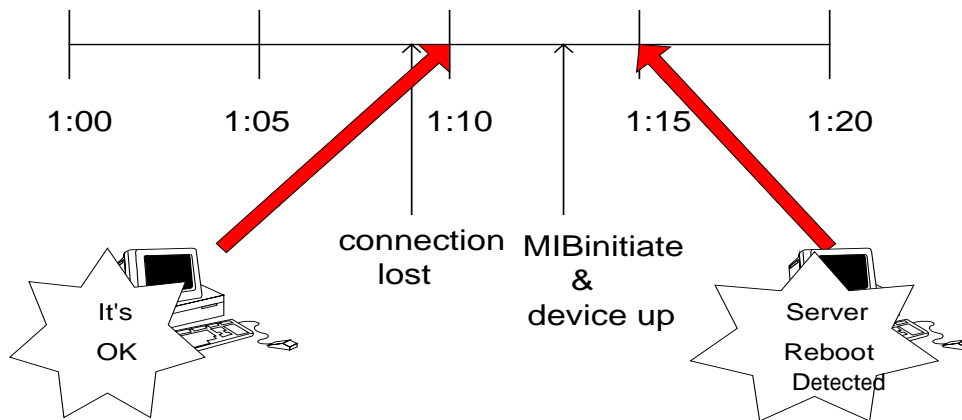


Figure 13. Fault Information Appearing in Two Adjacent Diagnostic Points

CONCLUSIONS

FJWorks is simply a network management tool experimentally developed by an academic laboratory. It is not trying to develop a sophisticated network management system to solve most of the network fault problems. The objective of this research is to develop a specific network management tool simply focusing on solving certain few but frequent appearing network fault problems. Based on this new idea, the system simply designed to solve five frequently appearing and well-known network fault problems. A network administrator may therefore promptly and effectively identify and resolve such faults appearing in the network.

The function of FJWorks basically consists of three main components including the Network Topology Management Subsystem (NTMS), the Network Traffic Retrieval Subsystem (NTRS), and the Knowledge-based Fault Management Subsystem (KBFMS). The *NTMS* is responsible for discovering the topology of the underlying network on which the system runs. It is implemented as an information base associated with the network topology. The network administrator can thus make use of the historic information stored in its topological repository to conduct further analysis thus achieving network fault prevention or avoidance. We believe this capability will provide network administrators having a more convenient tool to manage any network fault occurrence. The *NTRS* is responsible for collecting network traffic information and for identifying potential fault events.

It is implemented as distributed proactive agent-based technologies. The *KBFMS* is responsible for finding out all possible faults that occur in each network device according to the alarms received from the NTRS. The network faults are reasoned according to a mechanism of network fault reasoning and detection

According to the experimental results and comparison, the architecture and function we implemented in this research tool definitely shows us a promising result in conducting the fault detection from the five-selected well-known fault problems. These fault problems may be more effectively solved than using a complete and sophisticated network management tool. We believe to have a small and beautiful network tool focusing on certain specific network fault problems may facilitate network administrators to effectively identify and solve such error-prone network faults.

The capability of this experimental network fault management tool is still very limited. It only shows can effectively solve a small group of network fault problems. To make this system become usable, its fault-solving capabilities should have a dramatic improvement and extension during its ongoing development. Firstly, the system should be extended to solve more widespread fault problems. Secondly, The system fault inference engine also needs to be extended and improved accordingly, so that the fault detecting accuracy rate and the scope of network faults may be increased. Thirdly, the system architecture may be also implemented in a more intelligent informative model by embedding a push algorithm once a potential fault is detected.

REFERENCES

- American National Standard for Information Technology, "Fault Isolation-Information Characterization X3T8-1994", Draft, 1994.
- Bouloutas, A. T., Calo, S., and Finkel, A., "Alarm Correlation and Fault Identification in Communication Networks", *IEEE Transactions on Computer*, 42(2/3/4), Feb./Mar./Apr., 1994, pp.523-533.
- Lo, Cbi-Chun and Chen, Shing-Hong, "Robust Event Correlation Scheme For Fault Identification In Communications Network", Global Telecommunications Conference, 1998.
- GLOBECOM, "The Bridge to Global Integration", *IEEE Transactions on Networking*, (6), 1998, pp.3745-3750.

- Chao, C.S., Yang, D.L., and Liu, A.C., "An Automated Fault Diagnosis System Using Hierarchical Reasoning and Alarm Correlation", *Internet Applications, IEEE Workshop*, August 1999, pp.120-127.
- Cronk, R.N., Callahan, P.H., and Bernstein, L., "Rule-Based Expert Systems for Network Management and Operations: an Introduction", *IEEE Transactions on Networking*, 2(5), Sep. 1988, pp.7-21.
- Black, D. P., "Managing Switched Local Area Networks A Practical Guide", Addison Wesley, Inc., 1998.
- Feit, S., "SNMP: A Guide to Network Management", McGraw-Hill, 1995.
- Lewis, L., "A Case-Based Reasoning Approach to the Management of Faults in Communications Networks", *IEEE Transactions on Networking*, 1993.
- Network General Corporation, "Expert Sniffer Network Analyzer Operations", 1996.
- Maxion, R. A. and Feather, E. F., "A Case Study of Ethernet Anomalies in a Distributed Computing Environment", *IEEE Transactions on Reliability*, 1(4), Oct. 1990.
- Oates, T., "Fault Identification in Computer Networks: A Review and a New Approach", *Computer Science Technical Report*, pp.95-113.
- Fuller, W., "Network Management Using Expert Diagnostics", *International Journal of Network Management*, 1999, pp.119-208.
- Stallings, W., "SNMP, SNMPv2, SNMPv3, and RMON 1 and 2", Third Edition, Addison-Wesley Publication, 1999.

分散式網路代理軟體為基之乙太區域 網路錯誤管理系統

邱瑞科

輔仁大學資訊管理學系

摘要

資訊電腦科技與 Internet 的蓬勃發展使得我們的生活、工作型態及企業經營產生了重大的改變。透過 Internet 無遠弗屆的資訊傳遞特性與及電子商務、網路銀行與視訊會議等各種網路應用的蓬勃發展，不但縮短了人與人之間的距離，並且改變了企業與企業之間的交易方式與效率，同時也提升了企業經營的效益及有效的全球市場的開拓，這些都必需有賴快速有效的網路方能達成。因此，維持網路的正常和穩定的運作就成為網路管理重要的課題。企業網路任何的錯誤所造成的停頓，都將造成企業的無可彌補的損失。

然而，面對目前快速發展、網網相連及網路設施多樣化的複雜網路環境，網路管理的課題真是千頭萬緒。網路錯誤問題的偵測、確定、隔離及事先的預防，已成為網路管理重要的課題之一。

本研究之 FJWorks 為一個以代理人為基所建立之分散式網路錯誤診斷系統特別為區段診斷為基礎設計的乙太區域網路錯誤管理系統，乃是在輔仁大學通訊與網路實驗室所發展的一個研究產品。雖然目前市面上有諸多類似的產品，但我們嘗試用創新性整合智慧型代理軟體與知識為基之方式來建立一個效能與效率更高的網路某些常發生問題之診斷。系統將錯誤診斷的知識與故障排除程序建構成網路診斷 / 故障知識模式，故只要接收來自網路上的警訊便可以隨時監控網路的行為，並且能在重大問題發生之前就能做出適當的預警與處理。我們的功能是希望針對常發生的五種主要的網路錯誤加以診斷及進行問題的分析。也因為我們將功能侷限在一定的範圍之內，因此它可以比一般現存的類似工具更有效率。深信一個小而美的工具可以提供網路管理者更有效的來辨別並解決較易產生的網路錯誤問題。

關鍵詞彙：網路錯誤管理，因果關係，錯誤診斷，簡易網路管理協定，管理知識庫

