Malaysian Journal of Computer Science, Vol. 11 No. 1, June 1998, pp. 22-31

FMS: A COMPUTER NETWORK FAULT MANAGEMENT SYSTEM BASED ON THE OSI STANDARDS

Norleyza Jailani
Department of Computer Science
Faculty of Information Technology and Science
National University of Malaysia
43600 Bangi
Selangor, Malaysia
Tel: 603-8292931
email: norly@sun1.ftsm.ukm.my

Ahmed Patel
Computer Networks and Distributed Systems Research Group
Department of Computer Science
University College Dublin
Belfield, Dublin 4
Ireland
Tel: 353-1-7062476
email: apatel@ccvax.ucd.ie

ABSTRACT

Deals with the management of network faults using management applications based on the OSI standards. Presents the problem domain and a prototype system that provides several fault management applications.

Keywords: ***Network Management, Fault Management, Open Systems Interconnection, OSI, Common Management Information Services/Protocols, CMIS/CMIP, Manager-Agent/Client-Server Applications***

1.0 INTRODUCTION

Network management has become the central issue of discussion in the design and maintenance of communication networks. This is partly due to the amount in managing these networks and the tasks involved which include managing of local and remote hosts (e.g. servers), networked devices (e.g. routers, bridges), local and remote system administration (e.g. operating systems, storage systems, user profiles), local and remote applications (e.g. databases, directory servers), the network management systems and the network connections covering all the above entities. Obviously, with such an all-embracing definition, network management becomes a difficult task and requires a sophisticated system that could oversee all the network operations at a given facility.

Most organisations have recognised the strategic importance of communication networks. While in most cases, a better control ensures a higher level of performance, an improper or inefficient use of current network management instruments and techniques makes the management of certain network communication systems almost impossible. These trends can only increase the importance and visibility of effective network management. The urgency of the above situation has driven many organisations to work with the International Organisation for Standardisation (ISO) for the definition of a framework for network management. The open system interconnection (OSI) network management is divided into five Specific Management Functional Areas (SMFAs). They are fault management, configuration management, accounting management, performance management and security management. These functional areas are commonly referred to as FCAPS. They compliment each other and co-operate in order to accomplish all network management tasks. In this paper, however, we concentrate on the fault management functional area.

Fault management is usually mentioned as the first concern [1] in network management. Its main role is to ensure high availability of a network. Hence, involving a procedure to anticipate and avoid network failures, and in the case where a failure cannot be avoided, the necessary steps are required to contain the damage and resolve the effects on the network. These steps include capabilities to automatically detect and notify the occurrence of a failure, isolate the true cause of the failure and correct the failure whenever possible. We have chosen to investigate and address fault management issues due to the slow progress of standardisation process [2], the lack of fault management applications for users [3], multiple components' failures that are in general intermittent and require specific monitoring and troubleshooting tools to isolate the true cause of the failure, the large volume of alarms received and information required to perform fault diagnosis, and the need to anticipate network failures which in turn can reduce the time between failures and their restoration.

2.0 OSI NETWORK MANAGEMENT CONCEPTS

There are several network management concepts defined in OSI, including managed objects, managers and agents, and the management information base. Fault management as an application of network management uses the same concepts.

2.1 Managed Object

OSI network management standards use the principle of *object modelling*. The managed object forms an abstract representation of a resource viewed from the perspective of management. It may, for instance, be a physical item such as a modem, or it may be logical such as a circuit connection. Managed objects are characterised by their attributes, the events they generate, the actions they perform and the behaviour they exhibit [4]. The attributes contain the information of interest for management purposes, while notifications model the events that can happen at the real resource level for generating event reports or logs. Operations or actions are the set of procedures that can be applied on the object. And finally, behaviour is responsible for driving the changes in the object resulting from the execution of operations. Managed objects are stored in a management information base (MIB). Hence, all physical and logical devices are monitored and managed by fault management applications as managed object classes.

2.2 Manager and Agent

Since a manager cannot effectively control or administer a large number of objects directly, it employs agents (also known as *agent managers* or *agent processes*) that control subsets of the managed objects. The manager and agent exchange management information using CMIS/CMIP which are the network management services and protocols provided by the OSI standards (Fig. 1). The manager exchanges information by sending requests to and receiving responses or notifications from the agent. On the contrary, the agent receives requests from and emit responses or notifications to the manager asynchronously.

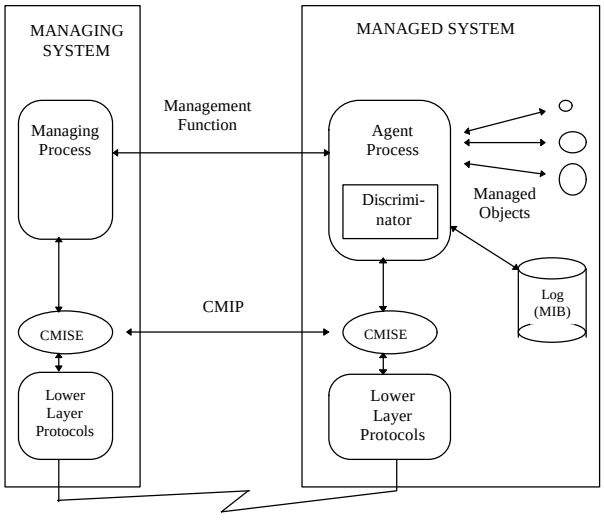

Fig. 1: General model for management communication

2.3 Management Information Base

The MIB contains an abstract image of the managed objects within a network, and provides a method of relating managed objects to each other. It is by manipulation of the virtual objects within the MIB that the actual resources within a managed system can be examined or changed by the network manager [5]. The MIB contains a virtual image of how a network is behaving at any time. Within the MIB, the managed objects are organised in a containment hierarchy or tree structure, where the subordinate nodes on the tree are contained within superior objects. This tree is called a management information tree (MIT). Each managed object has a name, and this name is formed by concatenating the names of all its superior objects within the MIT. The object at the top of the tree is the root and an object's name relative to the root is called its distinguishing name (DN). These names are used to select managed objects during the information exchanges that occur between managers and agents.

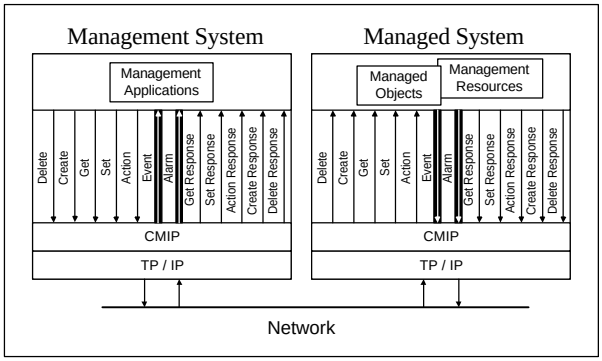

Fig. 2: CMIP working model: Interface with upper layers

2.4 CMIS/CMIP

The OSI standards management service named Common Management Information Service (CMIS) enables the access and manipulation of management information. The term service is used to represent a set of functions offered to a user by a provider, where the user is a management application or a higher level management protocol. The management service offers three types of use [6] which are monitoring, control, and reporting. These functions are realised through a set of primitives implemented by the OSI standard management protocol, called the Common Management Information Protocol (CMIP). The services offered by CMIS and provided by the underlying CMIP protocol are listed below:-

a) used by the manager application process

- **M-GET,** to retrieve attribute values from managed objects
- **M-SET,** to set up the attribute values of managed objects

- **M-CREATE,** to create an instance of a managed object
- **M-DELETE,** to delete an instance of managed object
- **M-ACTION,** to carry out other operations not included in the OSI standard
- **M-CANCEL-GET,** to cancel an M-GET request already in execution before its completion.

b) used by the agent application process

- **M-EVENT-REPORT,** to asynchronously report to the manager that some extraordinary event has occurred in a managed object.

They allow two OSI management service users to set up actions to be performed on managed objects, to change attributes of the objects, and to report the status of the managed object. Fig. 2 shows the outgoing path (downward arrows) of requests from the management system, across the OSI network to the managed system, or agent (upward arrows). The response from the managed entity (downward arrows on the managed system) is then sent across the network to the management system (upward arrows). CMIP spontaneously generates both alarms and event reports containing notifications about significant occurrences in the network.

2.5 OSI Network Management Entities

The OSI environment is made up of a complex collection of entities residing on a collection of seven layers, as laid down in the ISO Reference Model depicted in Fig. 3. Network management in the OSI environment includes management for each layer of the protocol stack, as well as overall management for the stack as a whole. These approaches to management, as well as the entities involved, are examined more closely in the section. OSI support for various network management requirements is provided in one of the following three ways. Firstly, as functions performed by the protocols themselves, secondly, as entities managing specific protocol layers, or finally as system management applications managing the entire stack of protocols, protocol-layer managers, and subnets.

The protocol suite provides direct communications and distributed processing support capabilities, and permits each protocol layer to monitor and control a single instance of communications within the layer. The system management application (SMA) such as fault management is an ancillary management facility needed to monitor, control and administer the entirety of the OSI resources that provide the communications and support capabilities of an open network environment. These five network management functional areas provide applications for a variety of decision making functions used to manage an entire OSI system (seven-layer stack). The SMAs are realised through a set of either centralised or distributed network management application processes which reside above layer seven. The OSI management framework provides three levels of OSI resource management:

1. Protocol management consists of those protocol-related internal mechanisms within any one of the seven protocol layers needed to control a particular communication instance. Protocol management is described within the standards describing each layer's protocol and services.
2. Layer management can affect multiple communication instances. It consists of those activities needed to manage all OSI resources associated with a particular protocol layer.
3. System management consists of those activities needed to manage the totality of OSI resources associated with any or all protocol layers within open systems.

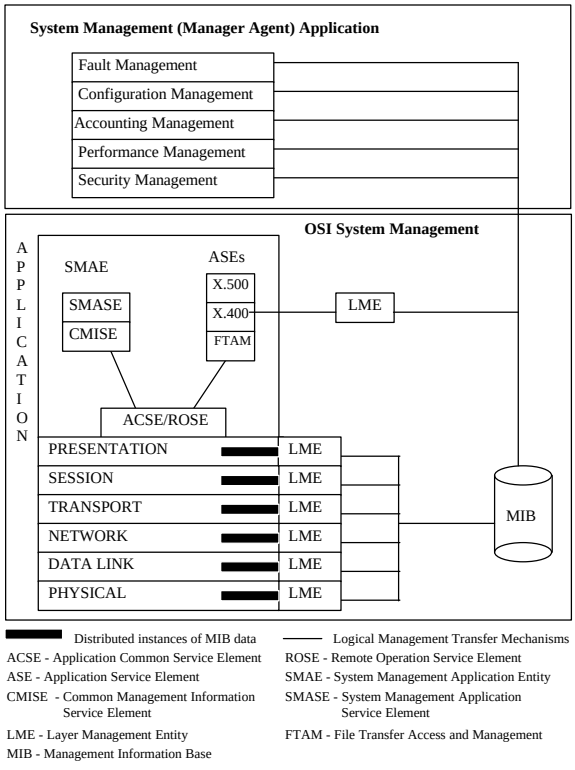

Fig. 3: OSI network management entities

3.0 FAULT MANAGEMENT - STATE OF THE ART

In this section we provide definition of some terms related to network faults and introduce the typical problems that may be encountered in performing fault management.

3.1 Definition Of Terms And Fault Characteristics

The terms *fault, error* and *failure* have often been confused. Incorrect interpretation of these terms may lead to their misuses. Definitions distinguishing them can be found in Wang's paper [7]. A *fault* is a software or hardware defect in a system that disrupts communication or degrades performance. An *error* is the incorrect output of a system component. If a component presents an error, we say the component fails. This is a *component failure. Failure* or component failure corresponds to the production of an error by this component. It is essential that we distinguish the terms described above. The *fault* is the direct or indirect cause of the errors. The *errors* are manifestations of the *fault.* The *failure* is the overall result of the *fault.* However, if a component produces errors, we cannot conclude that there is a fault within the component. Network faults can be characterised by several aspects such as their symptoms, propagation (transmission of an error from a component to another), duration in the network and severity [7]. Though network faults can be distinguished through their characteristics, it is worth noting that it is difficult to measure these properties accurately since they are subjected to the manner in which they are controlled and managed. The occurrence of a fault may be detected by users through symptoms that may be produced by some network components as a result of error. Fault symptoms can be associated to four types of error-timing errors, timely errors, commission errors and omission errors. These symptoms may take one of the following forms:

1. *An output with an expected value comes either too early or too late*: This situation is due to a timing error. It is usually seen by users as a slow response or a time-out, when their applications are indirectly influenced by the effects of the faults.
2. *An output with an unexpected value within the specified time interval*: This situation is due to a timely error which usually indicates a minor fault in the applications or underlying software and hardware.
3. *An output with an unexpected value outside the specified time interval*: This is due to a commission error. If no response is produced, it is associated with an omission error. An omission error can be regarded as a special case of commission error. A commission error usually implies a severe fault has occurred in the network.

If these symptoms are observed, there is a possibility that a fault has occurred somewhere in the network resulting in the components to produce erroneous outputs (errors). A fault in one component may have consequences on other associated components. Besides failing its own system, it may produce errors which could be transmitted to other components and degrade other systems as well. In this way, a fault in a single component may have global effects on the network. This phenomenon is referred to as fault propagation. A fault which occurs in an isolated systems may not affect other systems because there is no interaction between them. However, when the isolated systems are connected together by communications, errors produced by a fault may travel in a packet to other systems. A component can fail as a result of faults within it and the erroneous input produced by faults in other components. This is one of the major characteristics of network faults. Media for fault propagation include parameter, data and traffic.

The duration of a network fault, though recognised as one of its important criteria is somewhat difficult to measure. This is due to three reasons. Firstly, a fault will not be perceived until it produces errors. Secondly, it may take a long time for a particular fault to be isolated. Finally, the effects of network faults may not be eliminated automatically when the faults are removed. They will remain for some time until the operation is completely restored. Therefore, network faults can only be generally divided according to their duration into three groups: permanent, intermittent and transient. A permanent fault will exist in the network until a repair action has been taken. This results in permanent maximum degradation of the service. An intermittent fault occurs in a discontinuous and a periodic manner. Its outcome will be failures in current processes. This implies maximum degradation of the service level for a short period of time. A transient fault will momentarily cause a minor degradation of the service. Faults of the first type will cause an event report to be sent out and changes made in the network configuration to prohibit further utilisation of this resource. For a fault of the second type, the severity of the fault may transfer from being intermittent to being permanent if an excessive occurrence of this kind of fault becomes significant. Finally, a transient fault will usually be masked by the error recovery procedures of network protocols and therefore may not be observed by the users. It is fundamental for a designer of a fault management system to have knowledge of fault characteristics. This is because not all faults will have the same priority. The fault management system designer will have to decide which faults must be managed.

3.2 Fault Management Process

Recent literature [8-18] suggests that a comprehensive fault management system is composed of monitoring, reporting, logging, trouble ticketing, filtering, correlating, diagnosis and recovery activities. This section attempts to discuss the domain in which the fault management system will operate. For the purpose of this discussion, we have chosen to divide the activities associated with fault management into four major categories, namely detection, isolation, correction and administration. Error detection provides the capability to recognise faults [19]. It consists

of monitoring and reporting activities. Information provided by monitoring devices must be current, timeous, accurate, relevant and complete. Reporting activity include investigation of critical criteria which require notification and a mechanism for report generation. It also involves determining appropriate destination for sending notifications.

Its purpose is to isolate the actual fault, given a number of possible hypotheses of faults. Testing may be the most appropriate way of isolating the fault at this stage. Isolation comprises four activities: filtration, interpretation, correlation and diagnosis. Filtration involves analysing management information in order to identify new faults or if the fault has occurred before, to update its count. Filtration discards management information notifications that are of no significance and routes applicable notifications to their appropriate destinations within the system. Important information contained in the event reports must be extracted [8]. Here the nature, structure and significance of the event report are examined. Important information such as the name of the event report which normally represents the predefined condition that was met and triggers the generation of the event report itself. Other useful information is the time when the event report was generated. This information is important when performing correlation activity so that we may distinguish which events are related, and which are not. Correlation proves to be helpful, when two or more notifications received are actually due to a single fault. Through correlation, some faults may be indirectly detected. Hypotheses can be drawn from alarm correlation giving possible causes of fault. The objective of the diagnosis process is to isolate the cause of a fault down to a network resource. Given a set of probable causes, the diagnosis process is carried out. It involves identification and analysis of problems by gathering, examining and testing the symptoms, information and facts.

Once isolated, the effect of the fault must be minimised through bypass and recovery, and permanent repair instituted. Where applicable, steps must be taken to ensure that problems do not recur. This procedure consists of three activities: reconfiguration, recovery and restoration. Reconfiguration or bypassing involves activating redundant resources specifically assigned to backup critical entities, suspending services, or re-allocating resources to more important uses. The objective is to reduce the immediate impact of the failure. This function may be in a mixture of manual, semi-auto and automatic procedures. It may be possible for a fault to recover before any reconfiguration attempt is made. This depends on the nature of the failure, the criticality of the service and the expected time required to recover/reconfigure. Once a fault has been rectified, the repair needs to be tested and the entity returned to service. This needs to be scheduled at an appropriate time and depends on the expected service disruption in doing so.

Fault administration service ensures that faults are not lost or neglected, but they are solved in a timeous fashion [8]. This involves monitoring fault records, maintaining an archive of fault information, analysing trends, tracking costs, educating personnel and enforcing company policy with regards to problem resolution. It consists of three activities: logging, tracking and trend analysis. Logging maintains a log of event reports on faults that have occurred in the network. This will be used for trend analysis, reporting and future diagnosis of the same type of or similar failures [20]. Tracking keeps track of existing problems and persons responsible for and/or working on each one, facilitates communication between problem solving entities and prevents duplicate problem solving efforts [21]. This includes prioritisation of open faults due to their severity, and escalation of fault isolation or correction processes based on duration and severity of the faults. The current open fault records need to be ordered according to a priority scheme such that the most costly, or potentially costly failures are timeously resolved. The whole activities may be accomplished using a trouble ticket system [17]. Important information includes the frequency of occurrence of a particular failure and how much down time the various users are experiencing. Trends may indicate a need to redesign areas of the communication environment, replace inferior equipment, enhance problem solving expertise, acquire new problem solving tools, improve problem solving procedures, improve education, renegotiate service level agreements [8]. In this project, all activities in fault detection and isolation procedures and some aspects in recovery and administration procedures are implemented.

3.3 Typical Problems

One of the most critical problem associated with fault monitoring as given by Dupuy [14] and Stallings [21] is u*nobservable faults.* In this situation, certain faults are inherently unobservable through local observation. For example, the existence of a deadlock between co-operating distributed processes may not be observable locally. Other faults may not be observable because the vendor equipment is not instrumented to record the occurrence of a fault. Other problems are defined by Fried and Tjong [22] as follows. *Too many related observation:* A single failure can affect many active communication paths. The failure of a WAN back-bone will affect all active communication between the token-ring stations and stations on the Ethernet LANs, as well as voice communication between the PBXs. Furthermore, a failure in one layer of the communications architecture can cause degradation or failures in all the dependent higher layers. This kind of failure is an example of propagation of failures. Because a single failure may generate many secondary failures, they

may occur around the same time and may often obscure the single underlying problem. *Absence of automated testing tools:* Testing to isolate faults is difficult, expensive and time consuming. It requires significant expertise in device behaviours and tools to pursue testing. Even such a simple task as tracing the progressions of packets along a virtual circuit is typically impossible to accomplish. This leads to empirical rules of operation as "the only way to test if a virtual circuit is up, is to take it down" [14].

The process of recovery typically involves a combination of automatic local resetting combined with manual activation of recovery procedures. Recovery presents a number of interesting technical challenges. Which include a*utomatic recovery as a source of fault:* Since most network devices or processes are designed to recover automatically from local failures, this can also be a source of faults. The problem in fault administration is the maintenance of fault reports log which has been made difficult due to the lack of functionalities for creating or deleting records. The logging facility is usually performed in a static manner, where logging characteristics are stagnant. Therefore, some important fault occurrences are unable to be recorded. If log attribute values can be changed, the logging behaviour may also be altered.

4.0 FMS: A FAULT MANAGEMENT SYSTEM BASED ON THE OSI STANDARDS

In order to overcome the problems detailed above, we have designed and implemented FMS. It offers three types of fault management applications as depicted in Fig. 4 namely the Fault Maintenance Application (FMA), the Log Maintenance Application (LMA), and the Diagnostic Test Application (DTA). These fault management applications work together with the OSI Agent to perform fault management tasks on the network resources. The OSI Agent serves as a fault-monitoring agent. It has the capability to independently report errors to FMA. It can also issue a report when a monitored variable crosses a threshold. This allows the FMS to anticipate faults. In addition, it maintains a log of events. These logs can be accessed and manipulated by LMA. DTA provides a set of diagnostic tests which may be invoked by the user. This facility is beneficial to the network administrator whenever there is any suspicion that some of the resources are not functioning as desired.

The paragraphs that follow explain how these facilities help in increasing fault management efficiency.

The FMA solves the problems of unobservable fault due to ill-equipped vendor equipment to record the occurrence of a fault. This is done by monitoring the real resource critical properties and when significant events involving these properties occur, these events are reported to FMA so that further investigation is initiated. FMA acts upon the receipt of these events by performing other fault management processes on them. These processes include event interpretation and filtration, event correlation, invocation of predefined diagnostic tests and initiating recovery process. Event or alarm correlation is used by the FMA to solve the problems of too many related observation. In addition, the reporting criteria is designed to be reasonably tight to reduce the volume of alarms received by the FMA. In accomplishing this objective, only events that require attention are reported. Thus, in the FMS implementation, event reports are equal to alarms. Nevertheless, the objective to anticipate failure is neither neglected nor sacrificed. Hence, a number of managed objects are designed to issue event reports when the monitored attributes cross thresholds. Automated testing is provided in FMA by scheduling the execution of predefined diagnostic tests for every event report that is received, so that the source of failure becomes apparent. Subsequently, recovery action pertaining to the diagnosis is carried out automatically.

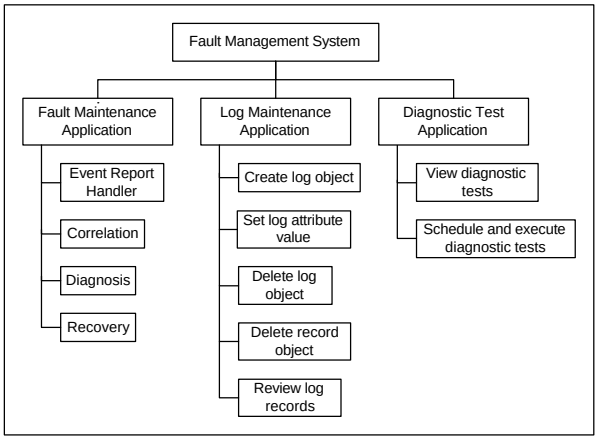

Fig. 4: The Fault Management System Architecture

On the other hand, the DTA provides a function that allows diagnostic tests to be invoked by the user. This facility proves to be beneficial to the experienced user who wants to skip trivial tests and choose only specific ones. This results in lower consumption of the network resources for the purpose of management. The lack of adequate tools for systematic auditing is overcome by the supports provided by the LMA. The LMA has the capability to initiate error condition (events) logging. Furthermore, the LMA can access these logs and control logging behaviour by setting their log attribute values. Other supports include facilities for deleting logs and log records, and reviewing events (by reviewing log records) for diagnostic purpose or trend analysis.

5.0 IMPLEMENTATION

The FMS was implemented on the OSI Management Information Service (OSIMIS) Management Platform. OSIMIS itself was developed at University College London (UCL) to provide a platform for implementing managed systems [23, 24]. OSIMIS provides OSI management protocols, a managed system implementation and run-time platform, as well as some simple management applications. The OSIMIS managed system platform is called the "Generic Managed System" (GMS). The supports provided by the GMS platform include provision of generic support for building OSI agents, provision of support for interacting with real resources, compilers to produce source (C/C++) code, some OSI client applications for the purpose of testing. To produce a managed system with the GMS, the implementor must implement C++ object classes that will be used to represent the managed object classes and the back-end to the agent to interact with the real resource. The implementation of the FMS has been divided into six functional components, as illustrated by Fig. 5, namely the User Interface, the Management Application, the System Manager, the System Management Agent, the Management Information Base and the Real-Resource Interface Programs.

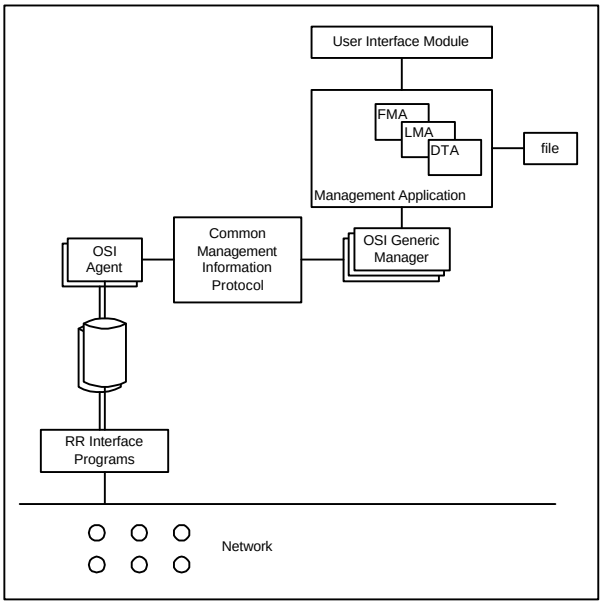

Fig. 5: FMS functional components

5.1 User Interface

Since the users of the FMS can have different responsibilities such as a network administrator or a network operator, the user interface provides a focused interface to the various fault management applications intended for different users. The network administrator is allowed to initiate or terminate the FMA. Fig. 6 shows the logical structure of application shell of the user interface if accessed from a generic network management system (GNMS) that provides all the five system management applications. This component has been developed using X Window application programming toolkits. The main logical entities of the User Interface are the FCAPS functional areas, the user of the FMS who initiate management functions, i.e. Administrator and Operator, and the various Management Applications and the facilities they provide.

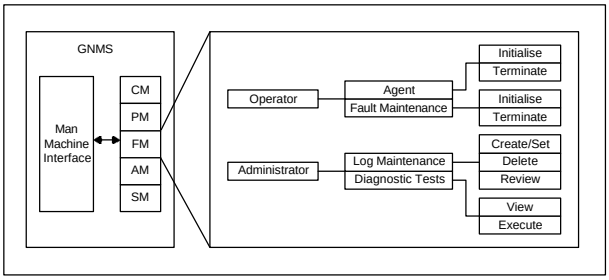

Fig. 6: User interface application shell

5.2 Management Application

The Management Applications have been developed in C++. There are three groups of Management Application, FMA, LMA and DTA. All of these applications are in executable form. The Management Application uses the OSI System Manager to send management operations to the OSI System Management Agent.

1. FMA

This management application consists of various management functions as described in Section 4.0. It deals with event reports from the moment they are received until recovery action takes place. Fault management procedures are realised through management function events handling, event correlation, fault diagnosis and fault recovery. It may be invoked by the user in an administrative role. In such cases, when the network manager's intervention is needed to perform a recovery action, output is given in the form of an advice or a suggestion. Otherwise, fault recovery is carried out automatically.

2. LMA

This logger accomplishes the task of maintaining a log of event reports. It is a tool for trend analysis and reporting, and provides library for future diagnosis of the same or similar problems [20]. LMA itself is realised in three separate programs. Facilities for logging and setting log attribute values are realised by program LogSet. Facilities for deleting log and logRecord are realised by program DelLogRec. The facility for reviewing logRecords is realised by program RevRec.

3. Diagnostic Test Application

This tool exists for the convenience of the user with the responsibility of a network administrator or technician. It allows the user to perform diagnostic tests on a particular network resource. Its purpose is either to check that the diagnostic tests provided function correctly or to perform a diagnostic routine as a result of reviewing the history of event logs recorded by a log at the SMA. DTA comprises two applications: *ViewTest* and *ScheExec*. *View Test* is the program that allows the list of existing diagnostic tests to be displayed. *ScheExec* allows diagnostic tests to be scheduled and executed.

5.3 System Manager

The OSI System Manager is an application which has responsibility for one or more management activities. It issues management operation requests (action) and receives notifications (events) on behalf of the Management Application function. Hence it comprises a set of programs which connect the OSI System Management Agent and retrieve the management information. These programs are the implementation of the CMIS primitives. They are used by the Management Applications for performing various fault management functions. The OSI System Manager sends commands to the OSI System Management Agent in CMIS requests form and waits for the replies. Five OSI System Managers have been implemented, namely the *cmis_set, cmis_get, cmis_evrecv, cmis_setlog* and *cmis_dellog* which perform the M-SET primitive, M-GET primitive, M-EVENT-REPORT primitive, M-SET primitive for logging event records at the OSI System Management Agent and M-DELETE primitive for deleting existing event records at the OSI System Management Agent respectively.

5.4 System Management Agent

The access to network elements defined for the MAs is through the agent. Access to a network element involves collecting its statistical information and manipulating it for management activities. The SMA is realised as a single UNIX process. SMA is the OSI agent support provided with the OSIMIS's GMS. SMA supports the ISO CMIP protocol. It performs scoping and filtering on the objects in the MIB instance and automatically handles all the CMIP functions. It performs management operations as a result of management operations communicated from the Management Applications. Its responsibility also includes issuing notifications reflecting the behaviour of the managed objects. SMA acts as the lowest-level functional components of the FMS and provides access to the X.25 network through the X.25 MIB. The tasks to be performed by an OSI SMA may be summarised as an internal communication with processes executing communication protocols, enabling remote access to managed object attributes through CMIS, enabling the external control of management functions such as the setting of thresholds within managed objects and detection of significant events and exceeding of thresholds, and the consequent generation of event reports.

5.5 Management Information Base

A MIB is implemented for each participating protocol i.e. X.25 MIB. The MIB communicates with the FMS through the ISO CMIS and CMIP. Facilities include the ability to both read and write specific managed object attributes, ability to add and remove elements to/from "set-valued" attributes, spontaneous generation of event reports and management of associations with remote manager processes. The managed object tree for a system varies as network activity varies. The implementation reflects this with objects being created and destroyed as necessary. The MIB is more than just a passive database. The implementation supports defined events which specify actions to be taken when specific network incidents occur. In the implementation, the action is always the sending of a report. Thresholds on counters and gauges are also supported. Exceeding a threshold may cause a report to be generated. The MIB is supported by a single UNIX process called System Management Agent as described in the former section. Access to these resources is realised via Real-Resource Interface Programs.

5.6 Real-Resource Interface Program

This is a part specific to a particular managed object representing a real resource. It must be tailored not only to the real resource type but also to the means the of how the real resource is used to present management information. It is not provided by the GMS and must be implemented by the MIB implementor. These real resources may reside in the operating system's kernel, on communication boards, in user-space processes or even at remote systems which are managed via a proxy management. Thus the interface programs vary according to management information from which real resources they want to access. At present, two kinds of interface programs are defined to extract information from the SunOS kernel and SunNet X.25 software for the X25 MIB. When an agent receives a CMIS request, it calls the interface program which executes the operation on a resource. Similarly, the interface program is called when a time-out indicating that real resource should be pooled. This process is carried out purposely to update real resource management information. As a result of processing information received from this interface program, a real resource managed object may determine that a threshold has been exceeded and a notification may be required. If the managed object determines that an Event Forwarding Discriminator filter is set to forward such a notification, it informs the SMA.

The applications provided by FMS have been designed and implemented in a modular manner and are independent of each other. This feature proves an advantage, in the sense that new applications can easily be added-on as and when their needs arise.

6.0 CONCLUSIONS

Most people are now depending on computers to get their jobs done. These mean that the computer networks must be available and offer good performance at most of the time. The occurrences of permanent and intermittent failures cannot be tolerated and must be avoided. Hence, proactive management of network failures is a significant item that must be included when planning a computer network. We have described the concept of OSI network management, and found that an important part of OSI management involved the network management applications (commonly known as FCAPS). The fault management functional area and its associated problems have been examined in detail. The conclusion is drawn that network fault management in such environment can be better achieved by providing supports of OSI-based fault management applications to assist network administrators and technicians perform fault management tasks. A fault management system has been suggested. The system offers various fault management facilities through three management applications.

The utilisation of OSI's CMIS/CMIP to request operations and receive notifications has proved the flexibility and richness offered by this service. The Connection-oriented mode of OSI's services support reliable transmission and are especially useful for FMA that has to receive notifications of attribute threshold crossings or fault occurrences immediately after these events are detected. Hence, the system is expected to promote the implementation of other management functional areas, as well as the utilisation of the OSI management protocol, thus promoting the notion of Open Systems.

REFERENCES

[1] K. Jones, "Managing Diverse Environments with CA- NICENTER for UNIX", *International Journal of Network Management*, May-June 1995.

[2] Paul J. Brusil, Daniel P. Stokesberry "Toward a Unified Theory of Managing Large Networks", *IEEE Spectrum* April, 1989.

[3] M. Jander, "Network Management Systems Get to Work", *Data Communications*, July 1991.

[4] ISO 10040, "Information Processing Systems - Open Systems Interconnection - System Management Overview", April 1989.

[5] A. Patel, "Security management for OSI networks", *Computer Communications*, Vol 17, No. 5, July 1994, p. 546.

[6] G. Mahamat, A. Das, G. V. Bochman, "An overview of Fault Management in Telecommunication Networks", *Advanced Information Processing Techniques for LAN and MAN Management*, J.-P. Claude (Editor) Elsevier Science Publishers B. V. (North-Holland), IFIP, 1994.

[7] Wang, "Model of Network Faults", *Integrated Network Management I*, B. Meandzija and J. Westcott (Editors) Elsevier Science Publishers B.V. (North-Holland), IFIP, 1989.

[8] F. Beck, "A Real Time Expert System for Enterprise Network Problem Management", Submitted in fulfillment of the requirement for the degree Master of Science (Computer Science) in the Faculty of Natural Sciences, University of Pretoria, Pretoria, August 1992.

[9] Abderrahman, "What's Needed for Distributed Systems Management", *International Journal of Network Management*, June, 1993.

[10] Joseph, J. Kindrick, K. Muralidhar, C. So, T. Toth-Fejel, "MAP Fault Management Expert System", *Integrated Network Management I*, B. Meandzija and J. Westcott (Editors) Elsevier Science Publishers B.V. (North-Holland), IFIP, 1989.

[11] Assoul, C. B. Westphall, "Management in Broadband Networks: ATM Networks Fault Management", *Advanced Information Processing Techniques for LAN and MAN Management*, J.-P. Claude (Editor) Elsevier Science Publishers B.V. (North-Holland), IFIP, 1994.

[12] Hong, P. Sen, "Incorporating Non-deterministic Reasoning in Managing Heterogeneous Network Faults", *Integrated Network Management II*, I. Krishnan & W. Zimmer (Editors) Elsevier Science Publishers B.V. (North-Holland), IFIP, 1991.

[13] Jakobson, M. D. Weissman, "Alarm Correlation", *IEEE Network*, November 1993.

[14] Dupuy, J. Schwartz, Y. Yemini, G. Barzilai, A. Cahana, "Network Fault Management A User's View", *Integrated Network Management I*, B. Meandzija and J. Westcott (Editors) Elsevier Science Publishers B.V. (North-Holland), IFIP, 1989.

[15] T. Bouloutas, S. Calo, A. Finkel, "Alarm Correlation and Fault Identification in Communication Networks", *IEEE Transactions On Communications*, Vol. 42, No. 2/3/4, February/ March/April 1994.

[16] Frontini, J. Griffin, S. Towers, "A Knowledge-Based System for Fault Localisation in Wide Area Networks", *Integrated Network Management II*, I. Krishnan & W. Zimmer (Editors) Elsevier Science Publishers B.V. (North-Holland), IFIP, 1991.

[17] L. Madruga, L. M. R. Tarouco, "Fault Management Tools for a Cooperative and Decentralized Netwrok Operations Environment", *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 6, August 1994.

[18] Leinwand, K. Fang, *Network Management: A Practical Perspective, John Wiley & Sons Inc., 1993.*

[19] CCITT, Management Framework Definition for CCITT Applications, Rec. X.700 The International Telecommunications Union, Geneva, Switzerland, 1989.

[20] Terplan, *Communications Networks Management*, Prentice-Hall (Englewood Cliffs, N.J.), 1992.

[21] Stallings, *SNMP, SNMPv2, and CMIP: The Practical Guide to Network Management Standards*, Addison-Wesley Publishing Company, Inc. (Reading, Massachusetts), 1993.

[22] Fried, J. Tjong, "Implementing Integrated Monitoring System for Heterogeneous Networks", *Network Management and Control.*, A. Kershenbaum, M. Malek, and M. Wall (Editors), New York: Plenum, 1990.

[23] MIDAS Work Package 2.2.1, "Runtime Support for Interaction with Real Resources in the OSIMIS management platform", Document MIDAS/ UCL/93/08, February 1993.

[24] Project INCA, "An OSI Network Management System", G. Night, G. Pavlou, Simon Walton, University College London, October 1988.

BIOGRAPHY

Norleyza Jailani graduated from Universiti Kebangsaan Malaysia in 1992 with B.Sc. (Hons). She received an M.Sc. in 1996 from University College Dublin for her thesis on computer network fault management system. She is currently a lecturer in Computer Science Department, at Universiti Kebangsaan Malaysia. Her research interests include network management, fault management, open systems, object-oriented design and framework.

Ahmed Patel received his M.Sc. and Ph.D. in computer science from Trinity College Dublin in 1977 and 1984, respectively. At University College Dublin he is a lecturer and head of the Computer Network and Distributed System Research Group. His main research interests include network management, security, protocol, performance evaluation, intelligent networks, CSCW and open distributed processing systems. He has published many technical papers and co-authored two books on computer network security and one book on group communication. He is also a member of the Editorial Advisory Board of the Computer Communication and Collaborative Computing journals.