World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:12, 2008

Design and Implementation of Rule-based Expert System for Fault Management

Su Myat Marlar Soe and May Paing Paing Zaw

Open Science Index, Computer and Information Engineering Vol:2, No:12, 2008 publications.waset.org/14911.pdf

Abstract**—**It has been defined that the "network is the system". This implies providing levels of service, reliability, predictability and availability that are commensurate with or better than those that individual computers provide today. To provide this requires integrated network management for interconnected networks of heterogeneous devices covering both the local campus. In this paper we are addressing a framework to effectively deal with this issue. It consists of components and interactions between them which are required to perform the service fault management. A real-world scenario is used to derive the requirements which have been applied to the component identification. An analysis of existing frameworks and approaches with respect to their applicability to the framework is also carried out.

Keywords**—**To diagnose the possible network faults by using the predetermined rules.

I. INTRODUCTION

FAULT management has traditionally been defined as the detection, diagnosis, and correction of a fault or problem. Typically the system is monitored to enable automatic detection. The implication (and in fact practice) is that there is a progression of events; first a problem is detected, then the problem is diagnosed, and then corrective actions may be taken. The issue with this approach is that it implies that a problem must be diagnosed before corrective actions may be taken.

Diagnosing a problem is very difficult because of the complexity of the systems and the constant change. The systems are complex functionally as a whole, and additional complexity is introduced through the interactions of heterogeneous components. The change comes about both internally through hardware and software updates and new components, and externally through environment. The behavior of the system is changing at different rates due to load, traffic mix and configuration. Typically diagnosis involves the correlation of information from a variety of sources: most of the approaches to correlation involve matching gathered information with known patterns or models of faults; at what level should information be correlated (depth), and how much information should be correlated – information is received at different times, and information from some sources may be incomplete or completely missing; in a distributed environment, systems are typically unwilling to share fault/performance monitoring data.

Some of common network management and supervision issues, such as alarm-filtering, automated diagnosis, and real-time aspects. Alarm-filtering is a feature that hides undesired event reports to the operator, to help him focus on more important information. Automated diagnosis can be achieved by the detection of known sequences of event-reports. As a matter of fact , some current network management platforms offer powerful tools, such as programming languages C, JAVA, to the operator to help him optimize and automate some tasks. The expert rules also have to take into account network related aspects such as the uncertainty of the date of the reception of an event [1].

Hardware requirements of this system are Intel Pentium - 4 - 2.8 GHz processor, memory 512 MB, and Hard disk storage space 40 GB. Software requirements of this system are window XP operating system, Apache or IIS web server, Microsoft Access 2003, J2SDK 1.5 or higher, JDBC Driver, JDBC ODBC Driver and Microsoft Access Driver.

Su Myat Marlar Soe Is With The Department Of Information Technology, West Yangon Technological University.
(E-Mail: Sumyat21 @Gmail.Com).

May Paing Paing Zaw Is With The Department Of Information Technology, West Yangon Technological University. (E-Mail: Maypaing25@Gmail.Com).

II. METHODOLOGY

A. Rule-based Expert System

Rule-based programming is one of the most commonly used techniques for developing expert systems. A rule-based expert system consists of a set of rules that can be repeatedly applied to a collection of facts. The following concepts are essential to rule-based systems:

- Facts represent circumstances that describe a certain situation in the real world.
- Rules represent heuristics that define a set of actions to be executed in a given situation.

There is a basic distinction between derivations and production rules. Derivation rules have the form if <condition> then <conclusion>, whereas production rules conclusion, in derivation rules, is abstract: it consists of deriving logical consequences from certain conditions. These logical consequences are simply asserted but not executed. An action, in production rules, is concrete: it consists of producing practical consequences from certain conditions. These practical consequences are concretely executed.

ECA rules, namely rules like if <condition> then <action>, which are production rules. However, production rules can implement derivation rules by using a special action "assert", which asserts knowledge. If the <action> part of a production rule is just a conclusion and not a function that performs

Open Science Index, Computer and Information Engineering Vol:2, No:12, 2008 publications.waset.org/14911.pdf

actions, we can consider this production rule as a derivation rule.

In both these cases, rules are composed of an if portion and a then portion. The if portion of a rule the left hand side (LHS), is called predicates or premises. The LHS consists of an expression, which can be a single expression (an individual fact that must be true for applying the rule) or a series of expressions (composite expression). In the literature of rule-based languages, a single expression is usually called pattern.

A composite expression consists of several single expressions connected together by using the conditional elements "and, or, not" in order to create complex rules. Usually, when several expressions are connected by the "and" conditional element, this element is omitted. In rule-based languages we have also the logical connectives " &, |, ~", which are used to manipulate values inside a single fact [2].

B. Architecture of Rule-based Expert Systems

The general architecture for rule-based expert system is depicted in Fig 1. The main elements of a rule-based system are facts, rules, and the engine that acts on them. The core of the architecture shown in Fig 1. consists of the working memory (fact base), the rule base (knowledge base) and the inference engine (rule engine) [3].

- The working memory contains facts that are the smallest piece of information supported by the rule engine.
- The rule base contains rules in the form of if-then statements, which represent the knowledge provided by the user and/or an expert of the problem domain;
-The inference engine matches facts in the working memory against rules in the rule base, and it determines which rules are applicable according to the reasoning method adopted by the engine.

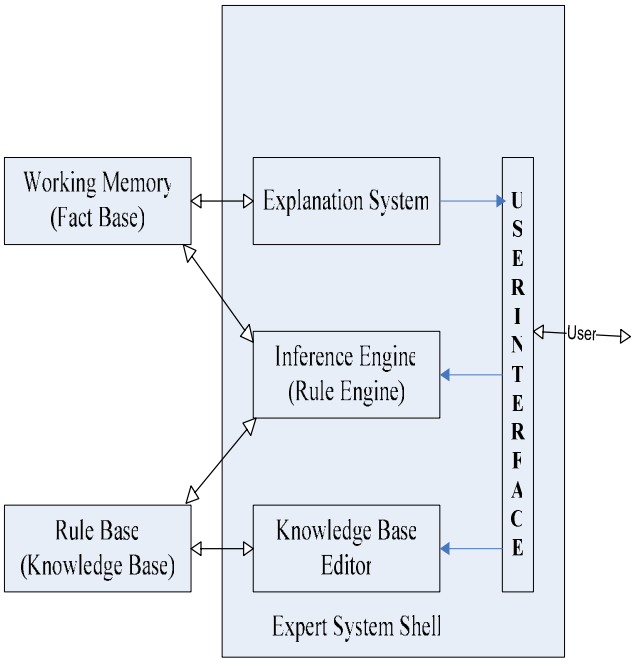

Fig. 1 General Rule-Based Expert System Architecture

III. Implementation on Fault Diagnosis

Main modules of the Network Diagnosis Rule-Based Expert System are Connection, Network Status Gathering System, Administration and Diagnostic. The relationships between each module are shown in the following Fig 2.

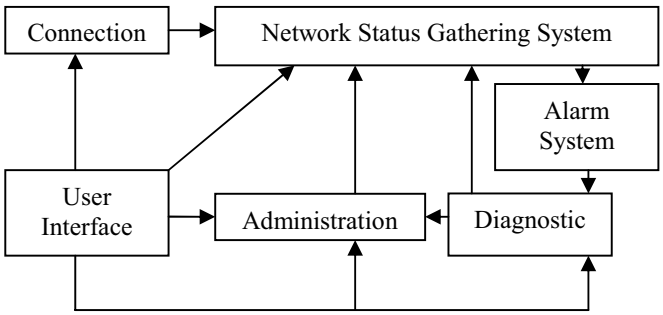

Fig. 2 Logical Components of the NDRBES.

User Interface is deal with all modules except Alarm System. Alarm system is background worker thread which will wait for an event to be occurred by time.

Connection module is used to check network availability and overall network status of the installed system. If network connection is enabled, Network Status Gathering System module will set up the event to detect on specific time. Events will be configured in Network Status Gathering System.

Administration tasks are to manage or set up an event to be an alarm. Which result of an alarm should notify to whom will be decided in this Administration task. Diagnostic is used to follow up the problems which encountered from the Alarm System's result.

Expert user (or) admin who handle the system will input configuration data to system. He will also make diagnosis with the system. System will present network information and alarm result to him. Network entities are network nodes which will be exist in the system network. System will check them according to user's request or schedule tasks. Their status will be response to system to record or report. System will report alarm result to Other Technical User who defined/configured by admin.

The overview flow chart of Network Diagnosis Rule-Based Expert System is as shown in Fig. 3 The following procedures are processing steps for network diagnostic rule-based expert system. The tester has launched the application after finishes all require installations. When application has started, system login form can be seen.

When tester typed password wrongly, it responses with an error message. After it has entered the valid password, main windows of the system is appeared. Before appear this main window, system load background worker thread of the alarm system according to configuration which had been entered by user last time. It means alarm system is enabled. If system is close, alarm system won't be existed in memory as alarm system is part of the system. When an error has occurred, user will be asked for making diagnostic When click the yes to

Open Science Index, Computer and Information Engineering Vol:2, No:12, 2008 publications.waset.org/14911.pdf

make diagnostic according to diagnostic tree, first inference of respective fault will be shown and ask for user response. When it has reached end of the inference, diagnostic result, suggestion for fault and possibility will be shown according to inference tree.

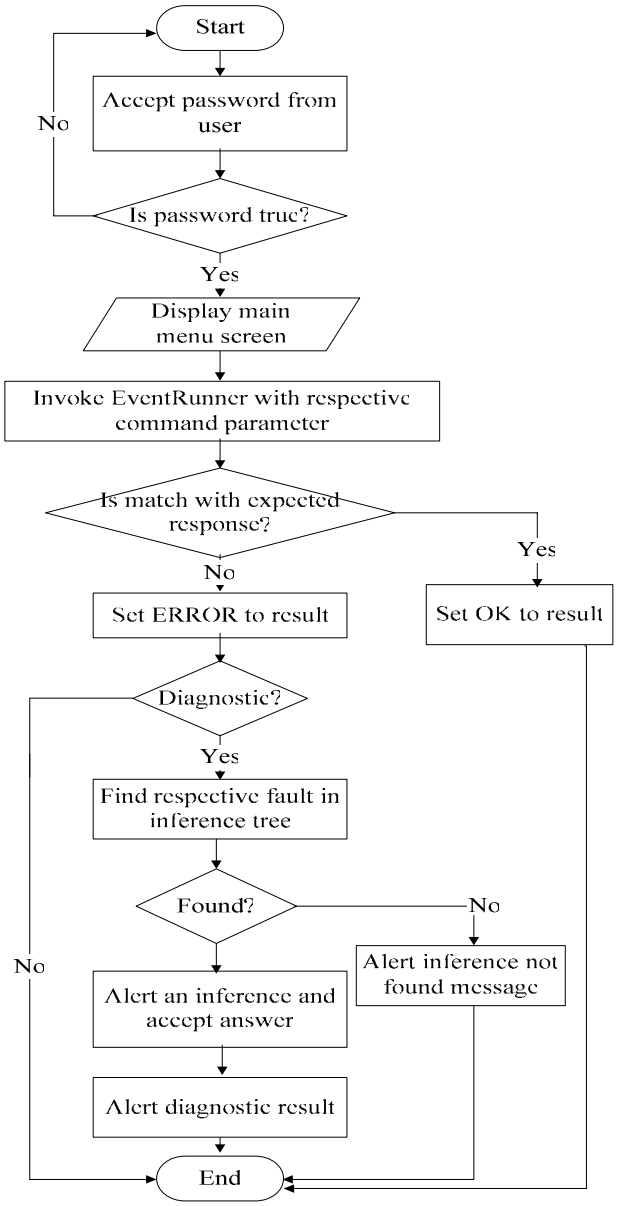

Fig. 3 Overview Flow Chart of Rule-based Expert System for Fault Management

IV. Results and Discussion

This section presents the detailed developments information of the Network Diagnostic Rule-Based Expert System. Java programming language is used to implement this system. It is robust, multi platform enabled, rich library, simple, etc. Net Bean is used to draw UI design efficiently and effectively. It reduces time consuming to draw UI in writing desktop application with java. Use light weight java thread to handle schedule tasks. Timer library from IBM is used to set up an alarm.

The software and development experiences along with testing provided by this thesis have been varied and interesting. It has given a taste of real world design. Most importantly, it has been learnt that attention to detail and systematic testing method s are the keys to a successful project. While the individual software modules (utility classes and UI classes) are developed unit testing is performed on these components using dummy calling modules.

Testing a system as a whole is complicated and takes a long time to cover all the possible paths and not feasible within a limited period the thesis can extend. Therefore, the testing is emphasized on the most important aspects. The project was also relied on careful design and verification process to reduce testing time.The main modules of this system are connection, Network Status Gathering System, Administrator and Diagnostic. The tester has launched the application after finishes all require installations. When application has started, system login form can be seen as follow:

Fig. 4 System Login

When tester typed password wrongly, it responses with an error message.

Fig. 5 Password is invalid.

A Test and Result of PC Network Status

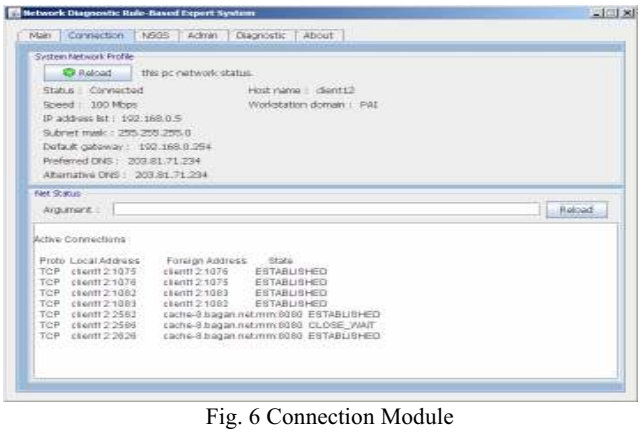

Fig. 6 Connection Module

Open Science Index, Computer and Information Engineering Vol:2, No:12, 2008 publications.waset.org/14911.pdf

As shown in Fig. 6, there are two main portions in connection module: Network Statistics Reloading and PC Network Status Reloading. To view all connections and listen ports the user clicks "Reload" to execute and set call back function for the operation when result is came back to program stack. If any error occurred in fetching network statistics with user inputted parameter, the error will show to user in text area instead of result. To view all IP configuration of host PC, the user can be clicked "Reload" of PC network status in connection tab. After execution complete, the required fields are split from "ipconfig" results. And then, user interface is updated. The information such as PC name, domain name are executed and updated those in user interface.

Fig. 7 Reloading Network Status with argument

B. Test and Result of Network Status Gathering System

This testing is implemented to query event list from data base, bind to list model to draw in Event list view, and check time of all events to enable alarm. If not, alarm timer is prior to current time. Thus the alarm is needed to set with the desire time from event.

There are four functions in selected event such as running, updating, deleting and rising alarm. As shown in Fig. 8, user can change event in event list view, prepare SQL statement to query the information of selected event. The query and fetch information are executed from database. The user interface is rendered with fetched information. And, also the Event History Grid is updated according to selected event.

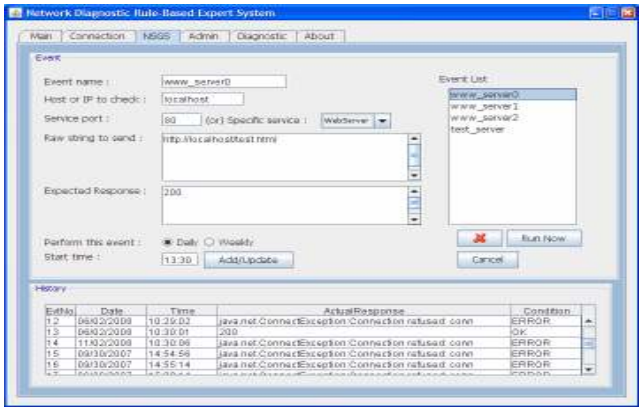

Fig. 8 View Event Detail in NSGS

When NSGS (Network Status Gathering System) runs, it can catch the event run result and compare with expected response. As shown in Fig. 9, if result is matched, database and user interface are updated and then alarm event rise in NSGS. Without selecting any event from user, the error evolves as shown in Fig . 10. The alarm rise and alert the user to be diagnosed. If user runs diagnostic, the respective fault can find and insert new event record in history.

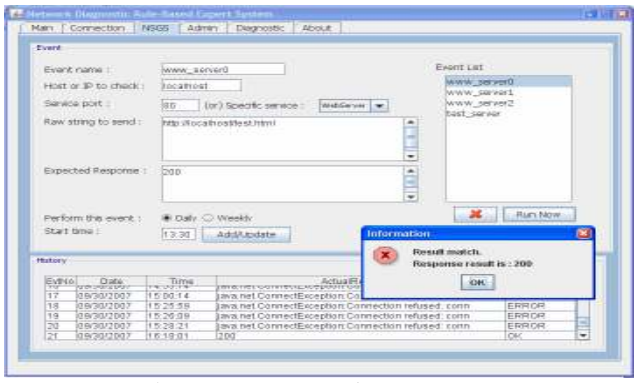

Fig. 9 Event Run Result – OK

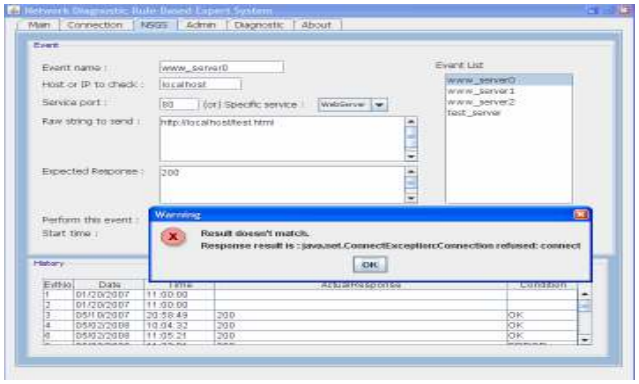

Fig. 10 Event Run Result – Error

C. Test and Result of Administration Facilities

The parameter "-a" in command window "arp" executes the current ARP (Active Resolution Protocol) entries after interrogating the current protocol data.

Fig. 11 Admin Module Main Screen

The ARP data is put in Generic Table Model to bind with JTable. As shown in Fig. 11 accepting new system password

Open Science Index, Computer and Information Engineering Vol:2, No:12, 2008 publications.waset.org/14911.pdf

for updating in database and searching the specific record according to date range are included in this testing.

By clicking "view" button, ARP (Active Resolution Protocol) viewer can appear. On entire network, the PCs information will be bind into Generic Table Model. The proxy setting from database can be loaded to display in respective fields. The event list in respective fields. The event list is loaded from database and bind those into list model.

D. Test and Result of Diagnosing Network Fault

In diagnostic module, SQL strings are prepared to query inference tree data. These data are structured as tree model as shown in Fig. 12 and bind to Jtree.

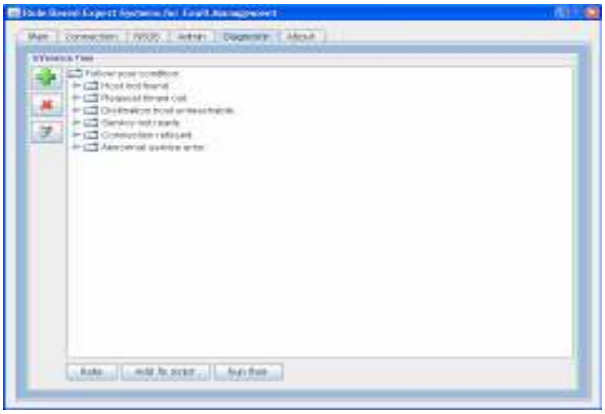

Fig. 12 Diagnostic Module Main Screen

To run the diagnostic module, the node is selected firstly, and then the fixer path can be obtained from the selected node. The fixer script is run to capture the result. It can be seen in Fig. 13. When user adds or updates fixer script in diagnostic tab, the selected node is obtained from memory. If it is adding a new node, the tree model can be updated (inserted) as a new node under selected node. After updating in tree model, database and user interface are also updated. Tree model is a temporary (memory) store place to hold tree diagnostic information.When user delete a node in diagnostic tab, the shallowest path to loop is necessary. If found a child, it is needed to go till leaf and deleted and loop to up. After all, the tree model is updated to database.

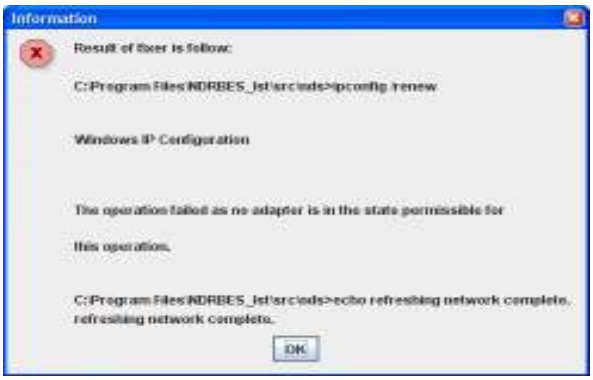

Fig. 13 Executed fixer result

V. Conclusion and Recommendation

In this paper we have shown the challenges that arise in the area of fault management which have been motivated by a real-world scenario. Components have been identified which are needed to address these issues and have been put into a service fault management framework. An implementation for large-scale services is tested. In doing so, a quantification of the framework's benefits will be addressed.

This paper proposes support for the design and implementation of a Rule-based expert system for fault management. Rule-based systems emulate human expertise in well-defined problem domains by using a knowledge base expressed in terms of rules. We have investigated the mapping of ECA-DL rules onto a well-known tool for developing rule-based system.

This work is an important part of the design and implementation of rule-based expert system. Our efforts included: (i) the study of expert system (ii) a literature survey in network fault management, (iii) an extensive study of rule-based systems, (iv) the definition of criteria in order to choose a tool for developing rule-based systems, (v) the extensive study of the chosen t tool and (vi) the definition of guidelines to allow ECA-DL rules. We have explored the benefits of using Event-Control-Action (ECA) pattern, since it provides a high level structure that helps in the design of rule-based expert system. This pattern divides the tasks of gathering and processing context information (*Event* module), from tasks of triggering actions in response to context changes (*Action* module) under the control of an application behavior description (*Control* module).

The ECA pattern reflects the reactive nature of network fault diagnosis whose behaviors can be expressed in ECA rules. In order to facilitate the execution of ECA rules by available technologies, we have made use of a specific language to define ECA rules. In order to find a suitable technology to execute ECA-DL rules, we have extensively studied rule-based systems. Rule-based systems emulate human expertise in well-defined problem domains by using a knowledge base expressed in terms of rules.

Acknowledgment

Firstly, the author would like to express her special deep gratitude to Dr. Khin Mg Aye, Rector, and West Yangon Technological University for granting to perform this thesis.

The author is greatly indebted to her chairman Dr..Htun Htun, Lecturer and Head, Department of Information Technology, West Yangon Technological University for her supervision, suggestion and constructive advice.

The author would like to express her deepest gratitude to Dr. Aye Aye Nyein, Assosiate Professor, Department of Electronic Engineering, West Yangon Technological University for her kindly helpful suggestion and advice.

The author would like to express her deep sense of gratitude to Daw Khin Kyu Kyu Win, Lecturer, Department of Electronic Engineering, West Yangon Technological

University for her kindly helpful suggestion, advice and encouragement.

The author also wishes to extend special thanks to Daw Thi Thi Soe, Lecturer, Department of Electronic Engineering West Yangon Technological University for her helpful suggestion to complete this paper perfectly.

The author would like to express her deepest gratitude to her external examiner, Daw Lei Lei Yi, Deputy Director, Myanma Scientific and Technological Research Department, for her interest and guidance.

The author especially appreciates and thanks all her teachers for their support and guidance during theoretical study and thesis preparation.

Furthermore, the author would like to express her indebtedness and deep gratitude to her beloved parents, for their kindness, support and understanding during the whole course of this work and encouragement to attain her ambition without any trouble. Finally, the author would like to thank all people who helped directly or indirectly towards the completion of this thesis.

Open Science Index, Computer and Information Engineering Vol:2, No:12, 2008 publications.waset.org/14911.pdf

REFERENCES

[1] Prem Viswanathan, Automated Network Fault Management, Master Thesis, CSHCN M.S. 96-3 (ISR M.S. 96-14), University of Maryland <http://www.isr.umd.edu>

[2] Riley.G..CLIPS, A Tool for Building Expert Systems. <http://www.ghg.net/clips/WhatIsCLIPS.html#ExpertSystems>

[3] Laura Maria Daniele Towards a Rule-based Approach for Context-Aware Applications <http://awareness.freeband.nl>

Su Myat Marlar Soe (M'06) became a Member (M) of MES (MYANMAR ENGINEERING SOCIETY) in 2006. The author was born at Ahlone Township Mon Yangon, Myanmar in 5th October 1982. The author has got Bachelor of Engineering in information technology, Thanlyin Technological University, Yangon, Myanmar, 2006 and Master of Engineering in information technology, West Yangon Technological University, Yangon, Myanmar, 2008.

She used to work as an Administrative Assistant at Myanmar Inspiration, Online Testing Centre, from January 2008 to September 2008. Now, she is a Ph.D student of Department of Technical and Vocational Education, Myanmar. She is deeply interested in network problems and how to solve them.