A New Network Management System with Ontology-supported Multi-Agent Techniques

Sheng-Yuan Yang
Dept. of Computer and Communication Engineering
St. John’s University
Taipei, Taiwan
ysy@mail.sju.edu.tw

Yi-Yen Chang
Dept. of Electrical Engineering
St. John’s University
Taipei, Taiwan
96N05018@student.sju.edu.tw

Abstract—This paper presents a new network management system with ontology-supported multi-agent techniques. This technique derived from the ontology combining with related free software — Ethereal and Cacti which stored the operating information of network management perfectly into the backend database; furthermore, from the view of the fundamental functions of information agents, the system could sketch the four main components of network management systems with the technique of graphic monitoring multi-agent, which the system architecture consisted of an interface agent, a proxy agent, a monitoring agent, and a search agent. The experimental outcomes of the system prototype proved that the techniques implemented in this paper could not only precisely recognize error alarms but also indeed reduce the recovery time to 61% of traditional processing time for network troubleshooting.

Keywords—Intelligent Agents, Graphic Monitoring Systems, Network Management, Network Flow Fetching.

I. INTRODUCTION

Nowadays, along with popularity of application and use of network technologies, it increasingly made network be complicated and enormous. How to effectively manage various network segments and equipments, understand their problem symptoms and accordingly bring up corresponding advices with the intuitively graphic interface at the right moment so as to promote network service quality and performance has become a very important challenge in the modern network management [10].

This study focuses on how to effectively integrate different networking devices under various enterprise environmental demands to develop a network monitoring management system. First of all, we construct a series of practical monitoring diagrams using the Cacti software and external software script following ontology theory. This enables us to completely and precisely generate an entirety network data by analysis and integration with a distributed intelligent agent mechanism [3]. It also substantially reduces the loading of the backend server databases. Finally, we can display a high-quality quantification diagram for a dynamic network data with system webpages [7]. This provides an easy, detailed solution that responds to the user’s questions of network problems. It reduces the user’s software expense and work-loading for the network manager, directly and easily providing a dynamic network data for all users through the system webpage.

![Conceptual architecture of the proposed system](image)

Figure 1. Conceptual architecture of the proposed system

The major parts of this technique are: ontology, data integration, and proxy mechanism. We also constructed the four major parts of an efficient service network system with graphic monitoring multi-agent: interface agent, proxy agent, monitoring agent and search agent, as shown in Figure 1. The interface agent is the communication bridge between user and system which transferred their messages completely. It also provides the query result through the function setup of operation interface. The proxy agent acts as an intermediary role between the interface agent and the search agent in order to reduce the retrieval loading of the backend server databases. The monitoring agent immediately collected and gathered various data from different network devices, and then stored those data in the dynamic network databases with the ontology-directed format for conveniently access by the system, and outputting the monitor results directly to users. Finally, the search agent executed the network information gathering, considering both user-oriented and domain-related concerns with ontology-supported operation models. This is the final product of an active and intelligent network management system with multi-agent techniques.

The application domain of Network Performance Monitoring (NPM) monitored the base level structure of wide-area networks and local-area networks, together with network
hardware equipment and their network operation status. This not only enabled the network control staff to understand the problem status in time, but also conveniently gave users sufficient information and helped share related knowledge to shortening the resolution time for network problems. The results demonstrated that the techniques implemented in this paper can both precisely recognize error alarms and also reduce recovery time [9] to 61% of the traditional processing time for network troubleshooting.

II. BACKGROUND KNOWLEDGE AND TECHNIQUES

A. Ontology

Ontology is a philosophical theory that explores the knowledge characteristics of life and real objects and provides complete semantic models for solving the problems of common sharing and communication. Describing the structure of knowledge content through ontology can create a knowledge core for a specified domain that can automatically learn related information in regard to communication and assessment. Such a technique can even induce new knowledge. As a result, ontology is a powerful tool for constructing and maintaining information systems [17]. This paper adopts Protégé (described later) to construct our domain ontology.

B. Management Information Base

The Management Information Base (MIB) was constructed in the monitoring and monitored network equipment, which can be used to store various SNMP objects (described later). Because the different kinds of network equipment with various communication protocols have their distinctive network management models, each management model has own set of MIB objects, including an equipment’s network interfaces, routing tables, IP packet transmission and receiving, etc. In the system, we refer to this information as SNMP objects stored in the MIB Ontology DB with a canonical format so as to begin monitoring related management information of network equipment.

C. Protégé

Protégé [13] is an ontology freeware package developed by SMI (Stanford Medical Informatics), which is one of the most important and frequently used platforms for constructing ontology. It uses multiple components such as Protégé-OWL Class, Protégé-Properties, Protégé-Forms, Protégé-Individuals, and Protégé-OWL Viz to edit and make ontology and led knowledge workers to constructing knowledge management system based on ontology. Furthermore, users can transfer to different ontology formats, such as RDF(S), OWL, XML or directly transfer into a database with better support function just like MySQL and MS SQL Server.

D. SNMP

The SNMP (Simple Network Management Protocol) is a simple and easy communication protocol for network management, which is a standard Internet protocol. It provides both a common standard for various network equipment and data for network management. This data can then be read and used for monitoring by network management applications. The SNMP agent mainly takes care of the commands received from the network management software and reports the current operating status of the network equipment periodically.

E. Developing Tools and Techniques

Cacti [2] is a development tool with PHP programming language to collect data by SNMP. It is integrated with various tools, such as RRDtool, SNMP, MySQL, and Apache [8] to create the network monitoring functions with graphic representation, which can store related monitoring information and their time-series data. In the system presented here, we use the 0.8.7 version. The RRDtool (Round-Robin Database tool) is the OpenSource industry standard, high performance data logging and graphing system for time series data, which can transfer those data to corresponding figures and renew those figures dynamically [14]. MySQL is a small-scale relational database management system with open source code that was developed by the MySQL AB Corporation in Sweden [12]. Currently, MySQL is popularly applied for medium- and small-scale websites to reduce the cost of website construction because of its small volume, fast speed, and lower cost.

Ethereal, developed by Gerald Combs, is an open source freeware network analysis system. It is also one of the best analyzer for network communication protocol [5] since it supports both Linux and MS Window platforms and its architecture adopts the protocol tree method. Since it is open source, Ethereal can quickly refresh its communication protocols and support the packet extracted file formats exported from various software package. Finally, it can go through the graphic user interface to clearly represent all network management information. In the proposed system, we use the 0.10 version.

For this system, we employed the WinPcap tool built into Ethereal to define the filtering regular expression [18], including the core packet filtering, a lower-level function base of dynamic linking library, and the pattern of the high-level system function base, as shown in Figure 2. Therefore, it can directly access the packet application interface to classify communication protocols and storage records, providing improved classification time and precision. The basic architecture of WinPcap consists of three parts: a Netgroup Packet Filter, called NPF, and a network packet catching module at the kernel level; and two function bases of dynamic linking library at the user level, including a lower-level Dynamic Linking Library and a high level and operating system independent System-independent Library.
The Multi-Agent System is a distributed environment formed by many kinds of agents. Currently, the construction trend of several software or information systems is to use many kinds of agent programs. In this study, we employ the multi-agent technique to develop the intelligent network management system with the related performance monitoring and visualized diagram drawing and refreshing. The system can surf the Internet and free-will execute following established rules and authorized scope, thus assisting the consignor to carry out the network information searching, filtering, arranging, analysis, and presentation without time or space limitations.

III. SYSTEM ARCHITECTURE AND OPERATING PROCESSES

A. System Development and Process

Figure 3. System operating architecture

Figure 3 illustrates the system operating architecture, in which the backend system server is a Windows 2003 Server and IIS 6.0 [19]. The intelligent agent systems and their corresponding environments were developed with Java [16], which usually resides and constantly executes in the monitor hosts. We also used KQML (Knowledge Query Manipulation Language) to communicate with each collaborative agent, including communication, coordination, and division of cooperation so as to collect the dynamic packet communication protocol and corresponding data in related networks, and then store this information in backend server databases for constructing the domain ontologies. Finally, we used the freeware Ethereal and Cacti to coordinate with software packages RRDTool and Net-snmp to help develop various network traffic flow and status diagrams. Those diagrams can combine the monitor figure agent with the domain ontologies to support the flow statistic analysis of domain network and corresponding communication protocols and Internet protocol analysis [20]. The front-end client webpage can display related information and status with the PHP syntax that completed the system monitoring capabilities. This also simplified the system installation and its set-up process, substantially reducing the difficulty in practically deploying the system [11]. Detailed working procedures of related multi-agent role, mission, and completed group are described as below.

B. Knowledge Modeling and Construction of Ontology Database.

Two important phases of knowledge modeling are the ontology normalization and ontology construction. The Ontology DB of this system is an ontology-based knowledge base, which mainly contains the definitions of network communication protocol. This study uses the tool WinPcap to construct the Packet Sniffer for defining a regular expression. The sniffer can employ regular expressions to directly access the application interface and classify those communication protocols and their storage records to improve the punctuality and precision of classification.

Figure 4. Network protocol and corresponding packet content

The ontology database constructed in this study first gathers statistics and analyzes related concepts of network protocols and then constructs a corresponding ontology database, as shown in Figure 4. The system is based on those characteristic words and classifies their corresponding standard packet format according to the classification patterns to enable convenient monitoring information citation by the Search Agent. The system employed the Protégé OWL editor [4] to construct the MIB Ontology DB to finish the second stage of ontology construction, is shown in Figure 5. Finally, support of the XML file (conveniently fixing the semantic errors if necessary) was used to transform that constructed file into the backend server database for conveniently access by the related system components. In addition, the system was arranged in pairs of the ontology and match rules, including the network protocol, source and destination IPs, as well as source and destination communication ports, to support all of the agent operations.
C. Interface Agent

Figure 6 illustrates the detailed architecture of the Interface Agent and the relationship diagram among other agents. Inside the agent, its modules contain the User Login for confirming the user’s login process, the User Manager Program for managing the user’s function access authority, the Personalized Web for presenting the personalized webpage, the Painting Component Model for drawing related quantification figures of dynamic information, and the Webpage Processor for predictably loading and processing the related webpages of visualized figures.

D. Proxy Agent

This proxy architecture is intended to improve the functions of the traditional architecture and develop an active mechanism for the Proxy Agent. It consists of the four modules, as shown in Figure 7, including the Proxy Space, which is responsible for autonomously storing the prior retrieval information; the Refresher, which is responsible for automatically refreshing the system information according to the definition of time interval in the Data Display Model; the Data Display Model, which is responsible for precisely showing the system information; and the Scheduler, which is responsible for periodically retrieving that information from the system databases.

E. Monitoring Agent

This system monitors the traffic flow and network equipment as shown in Figure 8. The monitoring agent contains: the Tracing Scheduler, which is responsible for periodically taking the database reading; the Relationship Tracer, which is responsible for associating the network communication ports; the Data Analyzer, which is responsible for analyzing and retrieving the traffic flow information; the Filter Rule Model, which is responsible for doing the warning calculation; and the Rule Scheduler, which is responsible for periodically sending those monitoring results to the Interface Agent. First, the agent uses the Tracing Scheduler to periodically read the necessary information from the system databases. Because the information records of the system databases are very diverse, the agent must be supported by the Relationship Tracer and Data Analyzer to analyze and abstract the necessary information chunks from the different system databases, such as the interface communication ports of the network equipments, the corresponding traffic flow in the network, and the time that information occurs in order to integrate this information. The monitoring agent also is supported by the Filter Rule Model for the various kinds of Data Reading using the Relationship Tracer and Data Analyzer. This includes the Data Analysis and Diagnosis, Data Processing, Error Alarm, and Normal State.
F. Search Agent

The Search Agent was established in the SNMP core, which employed related domain ontologies and related function libraries of Cacti and Ethereal to observe the connection status of the overall network and its equipment. Figure 9 illustrates its complete architecture, including the Linking Status, the Packet Sniffer, the Data Gathering, and the Protocol Analyzer. The operations were divided into the packet collection and the traffic flow collection. The former starts using the function of communication port monitoring in the network equipment. It employs the Packet Sniffer that was produced by the packet gathering functions built-in the Ethereal to collect all transmission packets in the network. Finally, it analyzes the communication protocol and its corresponding IP with the support of the Protocol Analyzer. The latter collects the traffic flow information of the whole network with the support of the SNMP Get communication protocol of the Cacti [15].

G. Summary

The interaction processes and their goal summaries of this multi-agent system are described as follows:

1. When the Interface Agent receives a user query, the Proxy Agent plays the role of mediator between the Search Agent and the system databases, which deals with the information temporarily retained in advance. This can provide the proxy mechanism to enhance the efficiency of the query cache services and thereby clearly reduce the query response time.

2. The Monitoring Agent is responsible for assisting users in monitoring network information, which can execute the corresponding information services and retrieve related information. Its service description format can directly acquire the query information from the system databases. Once finds information matching the user requirements, it immediately transmits that information to the Interface Agent to dynamically present the network statuses.

3. The Search Agent obtains related network information from the monitoring equipment through the real network connections and that information will become the data resources to construct the system databases. The network status information is divided into two parts: the real-time monitoring information and the continuous historical information. They are classified and separately stored in the backend databases. The system employs the backend databases to collect and store related network information in advance and then provides the information resources to the Proxy Agent for related query cache services.

4. The active and intelligent network management system is a multi-agent system directed by an Interface Agent. The Interface Agent not only plays the role of communication bridge between the user and the multi-agent system, but also distributes the user query requirement to the system and accordingly assists the user in presenting the outcome of analysis on monitoring information. The user can both use the system for common queries on problems between the equipment and traffic flow, and also go through the system to make hierarchical queries, to present the analysis results at a higher level and/or the specific details for replying to the user [6].

IV. SYSTEM DISPLAY AND VERIFICATION

A. System Prototype

To obtain clear understanding of all network situations, the homepage of the system was designed in a hierarchical manner consisting of four elements: 1) the left-upper corner: the status diagrams of equipment; 2) the right-upper corner: the trend diagram of the traffic flow; 3) the left-bottom corner: the billboard diagram of IP traffic flow; and 4) the right-bottom corner: the traffic flow analysis diagram of protocol in the network, as shown in Figure 10. The second layer of this hierarchy is the detailed situation of the network equipment, including all related information of the network equipment, connection status, communication protocols, etc. The user used only the client-end browser to connect with the log-in webpage of the system and the system immediately entered the certification screen to request the user account and corresponding password, for browsing the related system webpages after the successful certification. The user cannot install any additional software to use the system.
Figure 10. Homepage of the system

B. Warning Monitor

In the Window operating system, the system can diagnose the network connection state and its quality with the ‘Ping’ instruction. This instruction checks the network connection state with the Echo function of ICMP (Internet Control Message Protocol) according to the RFC 792 (Request For Comments) regulation unquestionably. In this system, we propose the filtering condition to analyze the network traffic flow with a standard value consisted of the high elastic analysis on traffic flow, modularized monitoring parameters, and composite monitoring conditions [1]. The formula is shown as follows:

\[
\text{Average} = \frac{\text{Reply time (1) + Reply time (2) + Reply time (3) + Reply time (4)}}{4}
\]

The normal green state occurs while the average response time is less than or equal to 40ms; the red warning alarm occurs when the average response time is greater than or equal to 81ms or there was no response at all; and otherwise, the yellow warning alarm appears. This can be used by the system to determine whether or not the existing network performance is good, and the system can conveniently proceed to eliminate the network malfunction in accord with the average response time.

C. Correctness of Warning Alarms Determining

In the experiment, we evaluate the traffic flow warning alarm and packet collection and classifying work. The network topology of the experiment environment was a closed local area network. We gathered all packets of the Host at the information center of a Branch of the Bureau of National Health Insurance in Taiwan between May 8, 2009 and May 21, 2009. The system operated in coordination with the corresponding user information base built up in advance to analyze and understand the usage situations of related IP within the network region for providing the two-way, interactive, and real-time monitoring query system with the Web manner, displaying all monitored and analyzed results, as shown in Figure 11. In this figure, the system displays the distribution diagram of the traffic flow warning alarms at each time point. The X axis shows the date and time, while the Y axis shows the average response time in milliseconds for a round-trip between the system and the monitored Host. Compared with the advanced monitoring diagram of network traffic flow, as shown in Figure 12, the system can precisely show that all of above time points indeed had higher traffic flow that affected transmission efficiency of the monitored network. The precision rate of warning alarm reached almost 100%.
D. Time and efficiency on the system processing

![Comparison chart](chart.png)

Figure 13. Comparisons between traditional method and proposed system

In its early stages, the network malfunction processing was made through an oral report of the problems by the user. However, administrators often encountered unclear statements by the users, which made it difficult to determine the precise kind of network problems. Thus, they could only depend on the network equipment topology and the SOP to sequentially check step by step, which took a long time. To handle a large-scale network topology, the time needed to fix problems did not encourage the administrators to resolve them. Thus, the traditional processing time and efficiency cannot already satisfy current practical requirements. Since the network staff spends so much time to check network problems, they cannot analyze and deal with related network problems in time. In the two-week period required to introduce this system into the practice and testing environment, we trained users to use the system, including how to view the network equipment status and the traffic flow information so users could understand their own network environment. If network malfunction occurred, the user can use this system to understand what kind of problem caused this event. The network staffs can go through the system to determine what problem occurred in which network node, and then eliminate related network situations. Figure 13 combines three items of common problems in the two-week introduction period, including the network equipment malfunction, the network connection problem, and the problem of large traffic flow. To analyze processing time, the processing time of network equipment malfunction was reduced to 67% of the usual (30/45=67%); the processing time of the network connection problem was reduced to 50% of the usual (10/20=50%); the processing time of the problem of large traffic flow was reduced to 67% of the usual (20/30=67%); and finally, the average processing time was reduced to 61% of the usual. This system was deployed and operated easily and it precisely and effectively provided warning alarms when a network malfunction appeared.

V. CONCLUSIONS AND DISCUSSION

This study uses many techniques to develop an active, intelligent, and multi-agent network management system using free software to construct a centralized caching system based on the network traffic flow. The system obtained the information through the cooperation and coordination of the intelligent multi-agent software. In addition, the system also provided warnings after analysis to monitor and predict some possible error events among controlled objects in the network. This provides an active and intelligent network management system with ontology-supported multi-agent techniques based on free software. This technique derived from an ontology combined with related free software, Ethereal and Cacti, stored the operating information of network management in the backend database. It also integrated with the intelligent agent technique to effectively enhance and improve the network monitoring performance, and accordingly present related quantification figures of dynamic information. The preliminary experimental outcomes of the system prototype showed that the techniques implemented in this paper could not only precisely recognize error alarms but also reduce the recovery time to 61% of the traditional processing time for network troubleshooting. This can be of great help for both network users and administrators.

The adaptability of the proposed system is enhanced by its use of the open source code. Users can precisely realize and control abnormal or irregular phenomena to determine problems in either network equipment or network circuits using the easy graphic interface based on the periodic monitoring analysis by this system. This alleviates the work load for network staff, and also reduces the cost of related network maintenance and professional training. In the future, we plan to expand our system to an improved and easy interface that considers both operating and management levels. This can allow the network administrator to assign and handle all network equipment through the visualization interfaces to then display and print out analysis reports and monitoring figures.

ACKNOWLEDGMENT

This partial work was supported by the National Science Council, Taiwan, R.O.C., under Grants NSC-98-2221-E-129-012 and NSC-99-2623-E-129-002-ET, and the Ministry of Education, Taiwan, R.O.C., under Grant Skill of Taiwan (1) Word No. 0990045921.

REFERENCES


