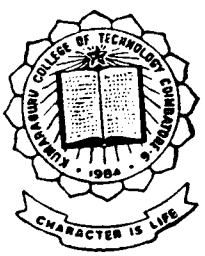
Network Analyser

DISSERTION SUBMITTED IN PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF COMPUTER APPLICATIONS
OF BHARATIAR UNIVERSITY

By
A. MUTHIAH
9438MO199



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
Kumaraguru College of Technology
COIMBATORE-641 006

June 1997

CERTIFICATE

This is to certify that this project work entitled

"NETWORK ANALYSER"

submitted to Kumaraguru College of Technology ,Coimbatore (affiliated to Bharathiar University) in partial fulfillment of the requirements for the award of the Degree of Master Of Computer Applications is record of original work done by **Mr.A.Muthiah** ,Reg N0.9438M0199 during his period of study in the Department Of Computer Science and Engineering,Kumaraguru college of Technology , Coimbatore under my supervision and guidance and this project work has not formed the basis for the award of any Degree / Diploma / AsssociateShip / Fellowship or similar title to any candidate of any University.

Professor and Head

Staff-in-Charge

Submitted for the University Examination Held On 3/6/97

Internal Examiner

External Examiner 21/6/97

CERTIFICATE

This is to Certify that the Project work titled

Network analyser

was carried out By

A.Muthiah

9438M0199

in partial fulfillment of the requirement for the award of Degree of

Master of Computer Applications

of

Bharathiar University

Coimbatore

during the sixth and final semester ended May 1997 under our guidance

Organization guide

Dr.D.Chandrasekar

Consultant

Telecom & Networking Group

DSQ SOFTWARE LTD

Institutional guide

Prof . P. Shanmugam

Head of the Department

Department Of CSE

Kumaraguru College Of Technology

Submitted for the sixth semester examination project work viva-voce held on \ / e /97

DECLARATION

I hereby declare that the project work submitted by me for the award of degree of **Master of Computer Applications** has not formed the basis for the award of any other degreee,diploma,association,fellowship of similar titles and this dissertation was done independently by me under the guidance of **Dr. D. Chandrasekar** and **Prof .P. Shanmugam** .

A. muthiah

A.Muthiah

DSQ
SOFTWARE LIMITED

TO WHOMSOEVER IT MAY CONCERN

This is to certify that the project titled **"NETWORK ANALYSER"** is the bonafide work done by **A . Muthiah** Final Year M . C . A . , Department of Computer Science & Engineering , Kumaraguru College of Technology , under my guidance during the period December 1996 to May 1997.

During this period, I found him technically sound,hardworking and very enthusiastic in his work.He is very receptive to new ideas and is able to implement them successfully.

I wish him success in all his future endeavours.

Dr.D.Chandrasekar

Consultant
Telecom & Networking Group
DSQ Software Limited

Organization Profile

Recognising the talent available in India for providing high quality ,cost effective software development services,many "FORTUNE 500" companies are outsourcing major projects to their Indian partners.

DSQ Software Limited ,ranked among the top seven software houses in India with a a initial investment of US $14mn , is an acclaimed "Centre of Excellence",with a perfect blend of State of the art and full range of computer facilities and creative human resources.

DSQ Software operates as a high quality and Low cost , software development in the niche markets of application development , software reengineering , maintenance and support.

DSQ Software provides services in the areas of technical applications using CATIA , CADDS 4X & 5X , I - DEAS , PRO - Engineer Business applications on IBM mainframes , AS / 400 , and Client Server platforms , Networking and Data communications.

DSQ Software ,is the youngest ISO 9001 IT company in India ,certified by KPMG Peat Marwick Quality Registrars , USA. It is commmited to International standard of quality and aims to acheive excellence in all its business processes.

DSQ Software has a team of over 800 software professsionals with combined experience of over 2000 person years in the IT industry.

Many multinational organisations such as MARCAM corporation and UNIFACE international and KANBAY Resources Incorporated have already established strategic

alliances with DSQ Software.

The hardware environment and the spectrum of services are shown in fig 1 and fig 2 respectively.

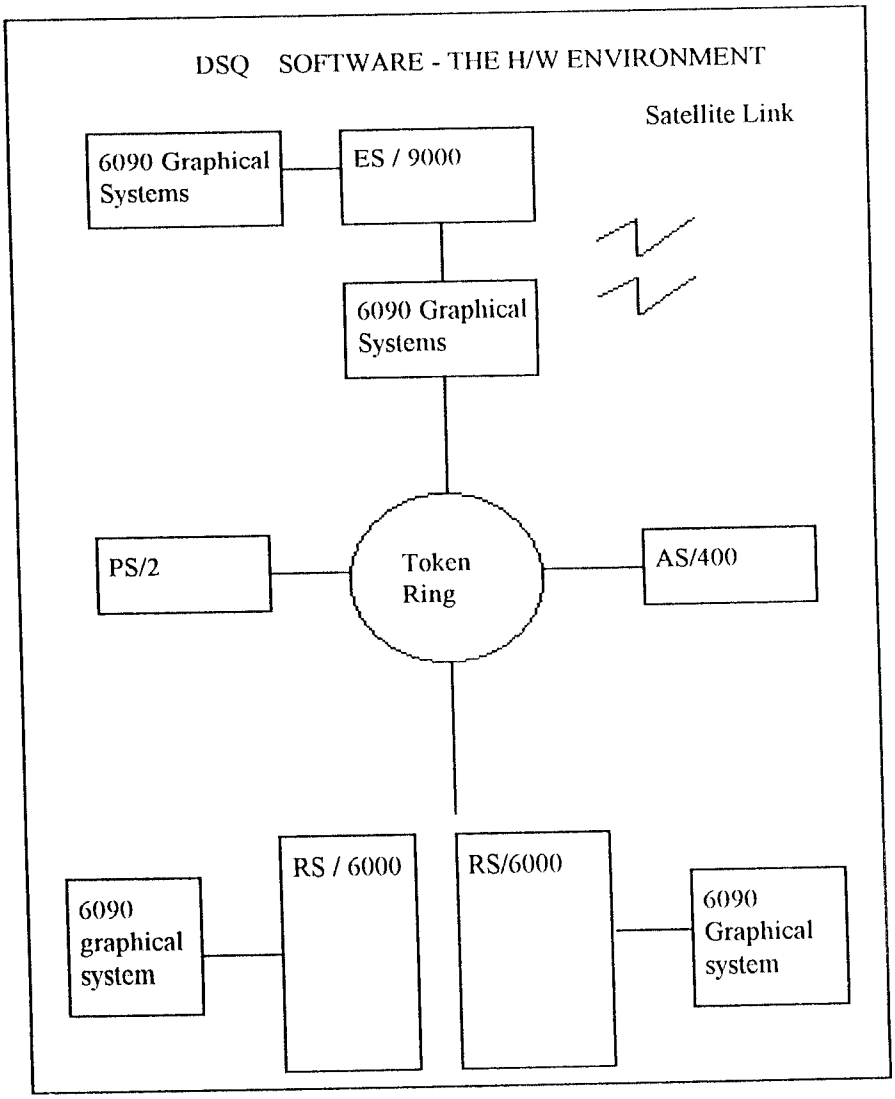


fig 1

DSQ Software - Spectrum of services

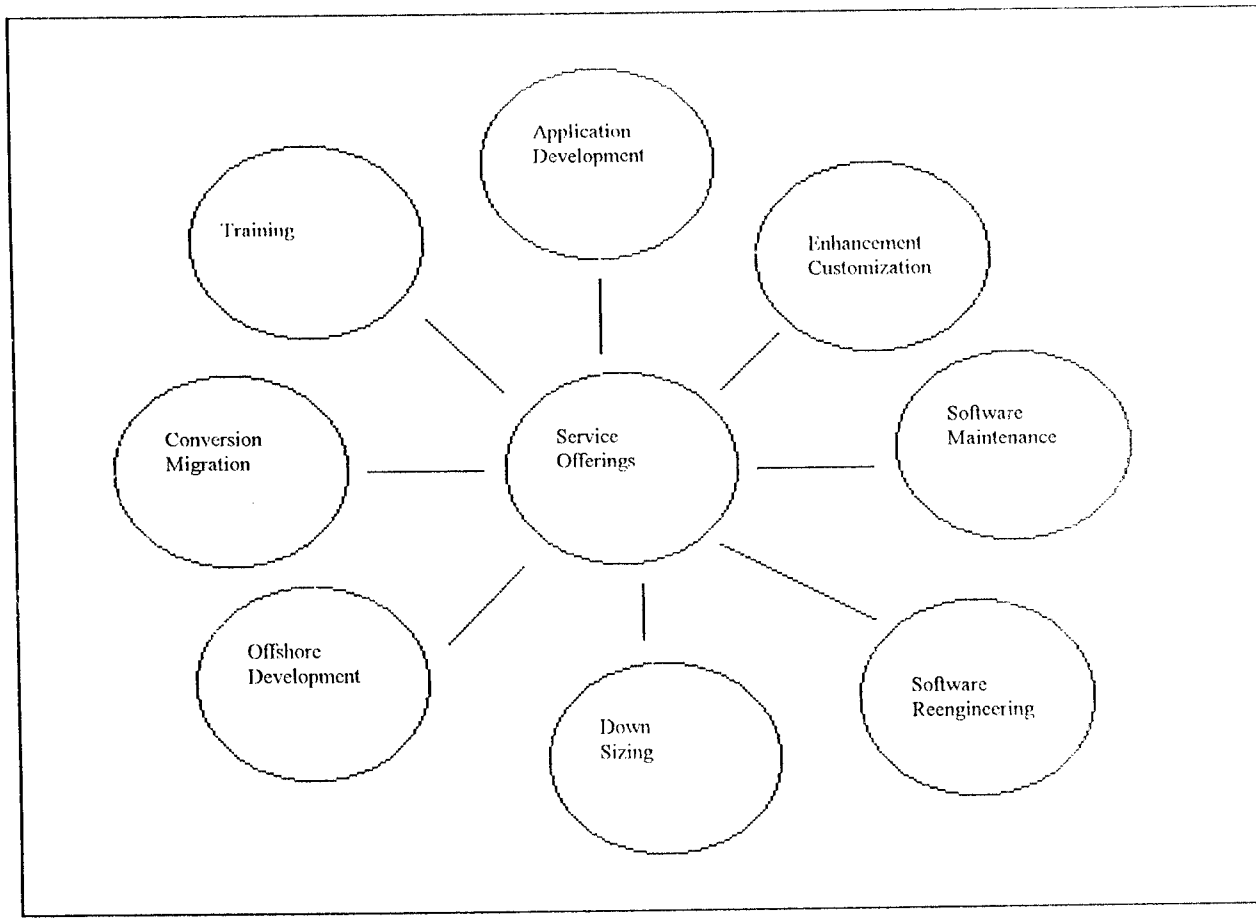


fig 2

ACKNOWLEDGEMENT

My sincere thanks to our respected Principal **Dr . S . Subramaniam** for helping me to undertake the project.

I am deeply indebted to **Prof . P . Shanmugam (Head of the Department)** for his invaluable guidance and support that has encouraged me to complete this project successfully.I am grateful to him for allowing me to do this Project with his constant help and support.

I am indebted to **Dr . D . Chandrasekar** , Consultant , Telecom & Networking Group, DSQ Software Limited , for his invaluable guidance , encouragement and timely help throughout the course of the project.

My deep sense of gratitude to **Dr . Harisekhar , Mr . Muthukumaraswamy, Dr . Vasuki Ashok** and **Mr.Murugan** for training me and imparting me with technical knowledge that enabled me to complete this project successfully.

I wish to convey my thanks to the staff members of Telecom & Networking Group, DSQ Software Limited,especially **Mr.Lakshman Prasad,Ms .Dhanalatchoumy, Mr .Manikandan,Ms. Meenalaxmi** for their kind cooperation extended to me.

I thank other faculty members of the Department of Computer & Engineering, Kumaraguru College of Technology.

I am greatly indebted to all my family members and friends for their support and encouragement.

SYNOPSIS

As computers have become smaller ,cheaper , faster and more numerous,people have become more interested in connecting them together to form networks and distributed systems. Data communication has become fundamental part of computing.

The merging of computers and communications has had a profound influence on the way computer systems are organised . Networks have revolutionized the modern computer world by making resource and information more shareable in reality. This has led to the proliferation of networks and as the networks increase in popularity the need for managing the networks to get the required functionality and high returns has increased.

The task of managing a network to provide a good and continuous service is a challenging one. Without an accurate understanding of how data is flowing through the network,efforts to effectively bridge or route traffic or otherwise coax the best performance out of the networks is based on guesswork . Difficulty in continuous managing of large , complex multivendor networks , increased the necessity of automated tools to analyse and manage networks.

This project ***Network Analyser*** aims in developing one such tool which would take the guessing out of performance tuning work by reporting on and storing network activity for later analysis and presents various parameters of the network in both statistical and graphical forms . The APIs and Front End Modules have been undertaken and completed successfully.

CONTENTS

PREFACE

he document starts with an introduction and the role or use of **Network nalyser** , the Network Monitoring System. With an idea of what ***Network nalyser*** is and it's role in Network Management, important modules of this roject and the modules implemented are explained in detail in the following hapters.

Chapter 3 presents the features of the development environment and the reasons for the choice of the same.

Chapter 4 gives an overview of the network monitoring system and the need for a new tool is explained. The chapter goes on to list out the desired features in the proposed system with particular reference the modules implemented.

In Chapter 5, the overall design of ***Network Analyser*** is described followed by detailed design and implementation details of the implemented modules.

Chapter 6 deals with the coding and testing of the modules. The programming style followed has been elaborated.

Chapter 7 concludes the report with suggestions and enhancements.

The Appendices contain insights into Networks, Dynamic Link Libraries, RMON and screen layouts. Abbreviations used in the document are also listed.

2 INTRODUCTION

NETWORK MONITORS

TWORK ANALYSERS , also referred to as network monitors or probes are ployed to study the traffic of a network as a whole. Network analyser is a work monitoring tool designed to give online status of the network activity of ious nodes in the network.

- It produces unbiased reports about multi - vendor products.
- It provides facilities to fine tune and forecast the performance of the network.
- It presents complete information about various parameters of the network in both statistical and graphical forms.
- The network monitor can be placed anywhere in the network and captures every data packet that crosses the node.

ypically, a monitor (or analyser)operates in 'promiscuous' mode viewing very packet on the LAN.

2.1.1 NETWORK ANALYSIS :

Network analysis involves monitoring the events of the network. The events are recognised by the values of some important parameters like utilisation, traffic, etc.

- The purpose of analysis is to assess the current 'health' of the network and to plan future course of action to be taken on the network to improve performance.
- The role of the network analyser is that it is the tool which the network administrator uses to decide on the action to be taken on the network . The monitoring system gives the network administrator complete information about all the parameters.

.2 ROLE / USE OF THE NETWORK MONITOR :

Network management is a collection of functions like

- Performance management
- Fault management
- Accounting management
- Configuration management
- Security management

Performance and Fault management require continuous monitoring of network activity. In network management , performance forecasting , fine tuning and error isolation functions play an important role for making future predictions.

Network Fine Tuning :

Growth in the network may lead to degradation in performance due to overload or errors. Hence the network administrator needs to ensure proper load balancing by choosing the poorly utilised server and adding nodes to it. The administrator may also delete some unwanted nodes to improve the performance.

Network Analyser can be used to obtain information about the utilisation of network by each node. This information helps the administrator in fine tuning the network and thereby avoiding performance degradation and keeping the network load balanced.

Error Isolation :

Various types of errors such as Cyclic Redundancy Check (CRC), Frame Alignment (FA) error, First In First Out (FIFO) error, etc. may occur on the Ethernet network.

twork Analyser helps the administrator in identifying the faults in the twork by conducting an error analysis to find the following :

- Dominant error on the network
- Highest error generating node
- Lowest error generating node

otocol Analysis and Distribution :

Protocol analysis (analysis of different protocols and their relative usage) done to determine the protocol distribution which indicates the relative loads used by each protocol on the network.

This information about the dominant protocol on the network is useful in lancing load on the network and also helps the administrator in taking protocol lated decisions that can be used to enhance the performance of the network as whole.

Features of the Environment

3 FEATURES OF THE ENVIRONMENT

his project has been developed on WINDOWS NT , a powerful GUI based etwork Operating System and an excellent platform for distributed applications. he API and front end have been implemented in Visual C++ , an integrated evelopment environment for developing windows programs using object oriented aradigm.

3.1 THE OPERATING SYSTEM - WINDOWS NT

WINDOWS NT is a complete Network Operating System which provides features additional to those present in Windows 3.1. It provides a graphical environment on a PC that offers a simple means of information exchange among many different applications.

The powerful features of WINDOWS NT include:

- Ability to run on fast RISC processors like MIPS and Alpha CPUs.
- Ability to support multiple processors.
- NT is a true pre-emptive multitasking operating system which overcomes the inefficiency of the non-pre-emptive nature of Windows 3.1.
- NT is multithreaded. This allows applications to multitask themselves and gives server applications the means to provide much higher throughput.
- NT's network services are part of the operating systemWith the software that comes out of the box,peer to peer networking is possible which allows stations on a network to share directories , printers ,and modems. There is built - in support for NetBEUI (Microsoft's network protocol), TCP/IP, Novell's IPX / SPX and IBM's Data Link Control (DLC). In addition to providing it's own network protocols,NT offers programming interfaces

for them. Named pipes, windows sockets, Remote Procedure Calls (RPC) and NETBIOS API are all supported.

NT allows machines to have multiple adapters supporting divergent network technologies,such as Ethernet ,Token Ring and FDDI,made possible by the standardised interface - the Network Device Interface Specification.

.2 HARDWARE DETAILS / REQUIREMENTS

lgents :

- Pentium 200 MHz with 64 MB RAM
- Network Interface Card should be any ethernet card supporting promiscuous mode

Branch Managers & Proxy Agent :

- Pentium 100 MHz with 32 MB RAM
- Network Interface Card – ethernet card

3.3 PROGRAMMING LANGUAGE - VISUAL C++

Visual C++ is today's premier Windows programming tool for serious programmers. It is Microsoft's current answer to the demands of Windows programmers, and is an enormous improvement over it's predecessors. Now, finally , the programmer has a programming environment that is a real aid, rather than an impediment .Visual C++ is apt to deal with the complexity of Windows programming.

The Microsoft Foundation Class Library is an important part of Visual C++ and the core of the application framework . The class library consists of a library of C++ classes and global functions with source code included. Other Visual C++ components are - AppWizard, ClassWizard, AppStudio, Visual Workbench,the Compiler,the linker,the debugger,the source browser,and the

-line help. These are the tools that the user can use to construct the plications.

SUAL WORKBENCH AND THE BUILD PROCESS

ıe Visual Workbench is a Windows - hosted interactive development vironment . It automatically generates the make file , known as a roject file". As part of the project the user can save compiler and linker vitch settings as specified through a series of dialogue boxes. To generate e executable program, the user simply chooses the Build command from e Visual Workbench Project menu. Visual Workbench contains a useful xt editor Windows interface standards and user colour to highlight C++ ntax.

HE APPSTUDIO RESOURCE EDITOR

Visual C++ uses AppStudio to edit most resources. AppStudio's native file ormat is the ASCII Windows resource (RC) file format , and each project sually has one RC file with #include statements to bring in resources from ther sub directories. AppStudio can also process Executable (EXE) and Dynamic Link Library (DLL) files. AppStudio compares favourably to the est applications written for Windows. AppStudio can even edit it's own esources.

THE C/C++ COMPILER

The Visual C++ compiler can process both C source code and C++ source code. It determines the language by looking at the source code filename extension.A 'C' extension indicates C source code,and CPP or CX indicates C++ source code . The Visual Workbench 's Use Microsoft Foundation Classes (MFC) option determines whether the compiler uses the MFC Library include files.

THE LINKER

To generate an EXE file, the Visual C++ linker processes the OBJ files that the compiler produces.If theVisual Workbench option Use MFC is specified, the linker uses the MFC library file for the appropriate memory model.

THE RESOURCE COMPILER

The Visual C++ resource compiler operates in either compile mode or bind mode. In compile mode , an ASCII resource (RC) file from AppStudio is compiled into a binary RES file. In bind mode , the RES file is merged with an EXE file. If the RES file is updated, it can be rebounded to its EXE file without relinking.

THE DEBUGGER

The Visual C++ debugger is the first ever Windows -hosted C++ debugging environment. The debugger works closely with Visual Workbench to ensure that breakpoints are saved on disk. Toolbar buttons toggle breakpoints and control single - step execution. To debug a program, it must be built with the compiler and linker options set to generate debugging information.

APPWIZARD

AppWizard is a code generator that creates a working skeleton of a Windows application with features, class names , and source code filenames that are specified through dialogue boxes. AppWizard code is minimalist code ; the functionality is inside the application framework base classes.It's purpose is to get the user started quickly with a new application.

CLASSWIZARD

ClassWizard is a program (implemented as a DLL) that operates both inside the Visual Workbench and inside Appstudio.ClassWizard takes the drudgery out of maintaining Visual C++ class code.ClassWizard writes the prototypes, function bodies, and code to connect the messages to the application framework when

ıew class or a new function to handle a Windows message is needed.

E SOURCE BROWSER

e Visual C++ Source Browser lets the user examine an application from
class or function viewpoint instead of from the file viewpoint. The
›wser has the following viewing modes:

Definitions and references
Call Graph / Caller Graph
Derived Class graph / Base Class Graph

NLINE HELP

he entire contents of the Windows SDK reference manuals and the class
›rary reference manuals are included in the Visual C++ on-line help. Help
also available for AppStudio, AppWizard , and ClassWizard. Visual C++
lelp resolves conflicts between Windows SDK function names and identical
Microsoft Foundation Class Library names . If a function name is selected
hat corresponds to member functions in several classes ,the class can be
hosen from a list box. If the help is asked on a class, a member function and
lata member list will be seen in functional order.

4 SYSTEM STUDY AND ANALYSIS

4.1 INTRODUCTION

System study and analysis is about understanding situations, to learn how a system currently operates and to find where improvements should be made.The existing system is described briefly before proposing a new system.

4.2 EXISTING SYSTEM

The existing Network Monitoring System has been implemented in Windows 3.1 with front end implementation in Visual C++. It is a powerful monitoring system for a Local Area Network and is in itself , a successful and competitive Network Monitoring System. But some of the features that it lacks or requires amendments are:

- extension to larger networks or groups of networks
- monitoring of remote networks or nodes
- support of token ring, FDDI etc.
- a flexible and featured platform like Windows NT
- a good user friendly GUI

These features are expected to be satisfied in the proposed system.

3 THE PROPOSED SYSTEM

The proposed Network Analyser System, is a mega project proposed to support the functions described below:

SALIENT FEATURES OF THE NETWORK MONITOR :

The network monitor is expected to perform the following functions :

- Network statistics collection
- Intrasegment communication
- RMON (Remote Monitoring) support
- Packet related information
- Multi-process watch
- RDBMS support
- Protocol wise listing
- Node wise listing
- Error wise listing
- Application Recognition

Other functions supported are :

- Connectivity information
- Play back option
- Trouble shooting
- Bitmap and Icon based GUI

3.1 THE FUNCTIONS IN DETAIL :

NETWORK STATISTICS COLLECTION :

he network monitor captures all packets on the network and determines
ie values of the following network parameters :

1) Utilisation
2) Throughput
3) Traffic
4) Protocol traffic
5) Error percent

The values of the parameters indicate the health of the network.

NETWORK PARAMETERS IN DETAIL :

(1) Utilisation :

This parameter is a ratio that indicates how efficiently the bandwidth of the cable is being utilised.

It is defined as follows :

$$\text{utilisation} = \frac{\text{actual number of bits transmitted on the network at any instant}}{\text{capacity of the network}}$$

The contribution to utilisation by a node is defined as the ratio of the number of bits transmitted by the node at any instant to the number of bits transmitted on the network at the same time. This gives the relative utilisation which can be used for controlling or accounting purposes.

(2) Throughput :

This parameter, expressed in bits/sec, indicates the output from the cable at any instant. The throughput of the network can be defined as the number of bits that passed through the cable in one unit of time.

(3) **Traffic :**

Expressed in packets/second, traffic is defined as the number of bits passed on the network in a unit of time.

The contribution of a node to traffic (expressed in percent) , is the ratio between number of packets sent by the node to the number of packets in the network in the same unit of time.

(4) **Protocol traffic :**

This gives the relative loading of the network by various protocols and is defined as the ratio of the number of packets used by the protocol in question to the total number of packets on the network in the same unit of time.

(5) **Error percent :**

The different errors that can occur on an ETHERNET are :

- ***Cyclic Redundancy Check (CRC) error :***

When a packet arrives , the hardware calculates the CRC of the packet (including the fields Source address,Destination address,Type and Data fields). If this value does not match the value in the File Check Sequence (FCS), then this is called CRC error.

- ***Frame Alignment (FA) error :***

A packet that does not end on an 8 bit boundary is called a packet with a FA error.

- ***Jabber :***

A packet having length greater than the allowable maximum (1518 bytes on an Ethernet) with a bad CRC is called a Jabber.

- ***Runt :***

A packet having length less than allowable minimum of 64 bytes with a good CRC is called a Runt.

- ***First In First Out (FIFO) overrun :***

If the ring buffer in the network interface card (NIC) has got overrun,this error is reported.

rror percent = Number of packets originating from a node in a unit of time / total number of erred packets on the network in the same unit of time

All these parameters are useful in error isolation and cause analysis.

2. *INTRASEGMENT COMMUNICATION*

Communication between networks with different operating systems using different protocols is supported by the Network monitor. In other words, intranet communication is supported by the Network monitor.

3. *RMON SUPPORT:*

For effective network management, there would typically need to be one analyser per sub-network; the analysers (which may be workstations, servers or routers) need to communicate with the central network management station. In this context, they are called remote monitors.

e network monitor supports the RMON MIB groups namely :

Statistics - maintains low-level utilisation and error statistics for each sub-network monitored by the AGENT (each node in a network management system which includes an NME ... Network Management Entity i.e., a collection of software devoted to network management task).

History - records periodic statistical samples from information available in the statistics group.

Alarm - allows the person at the management console to set a sampling interval and an alarm threshold for any counter or integer recorded by the RMON agent.

. **Host** - contains counters for various types of traffic to and from hosts attached to the sub-network.

5. **HostTopN** - contains sorted host statistics that report on the hosts that top a list based on some parameter in the host table.

6. **Matrix** - shows error and utilisation information in matrix form, so the operator can retrieve information for any pair of network addresses.

7. **Filter** - allows the monitor to observe packets that match a filter. The monitor may capture all packets that pass the filter or simply record statistics based on such packets.

8. **Packet capture** - governs how data are sent to a management console.

9. **Event** - a table of all events generated by the RMON agent.

4. *PACKET RELATED INFORMATION:*

Information which includes the node name , number of broadcast / multicast packets transmitted by the nodes, traffic rate of broadcast/multicast packets etc. must be displayed to help the network administrator in restricting a user from transmitting too many broadcast packets in the network.

5. *MULTIPROCESS WATCH:*

The multi-threading concept of the platform, WINDOWS NT is exploited so that each thread has a process running.

6. *RDBMS SUPPORT:*

Since a large amount of data is involved , RDBMS is used to handle and manage the whole data. If a RDBMS is present, the administrator can know the value of a single parameter on a particular day or he can know the parameters of the network one year back . Hence , RDBMS allows efficient use and maintenance of data , allowing more data to be handled and easy storage and retrieval.

7. *PROTOCOL WISE LISTING:*

In a multi-protocol network, one has to know which protocols are most widely used. As server based systems are popularly used in a network, each protocol traffic can be viewed as a single server traffic. Hence any excessive usage of a particular protocol means a server which supports the protocol is being used widely. The protocol wise listing provides information such as number of packets committed,traffic percent, etc.of each protocol, which helps one to design or redesign to reduce the congestion at a server.

A graphical chart which compares the traffic rate of each protocol has to be displayed. The protocols to be supported by the Network monitor are :

- NOVELL
- 3COM
- TCP/IP
- ARP
- NETWARE LITE
- NFS
- NETBIOS

8. *NODE WISE LISTING:*

The network administrator may wish to know the content of each node in the work towards the network traffic and network performance. The Network nitor has to provide information such as :

- Active node list - statistical information about currently active nodes in the network
- Statistics about each node in the network
- Graphs which relate the utilisation and traffic rate between nodes in the network.
- Graph of the frame size distribution in the network.

9. *ERROR WISE LISTING:*

The Network monitor must help to identify the transmission errors in the cable and identify them as one of the following :

- CRC
- FA
- FIFO OVERRUN
- RUNTS
- JABBERS

The error table has to present the percentage of various types of error packets produced by each node and identify the percentage of error contributed by each node to the total error packets on the network. A comparison of the number of different types of error packets transmitted by each node in the network must be presented in graphical form.

10. *APPLICATION RECOGNITION:*

The network monitor provided application layer details through the service calls available in Windows NT.

11. *CONNECTIVITY INFORMATION:*

Physical layer details are provided by the Network monitor through the service calls available in Windows NT.

12. *PLAYBACK OPTION:*

The option provided to help the administrator to view the past network performance parameters like utilisation, traffic, error rates etc. for a period of half an hour. This will enable the administrator to compare the current statistics with the past statistics and provide the buffering of statistics.

13. *TROUBLE SHOOTING:*

The concept of intra-segment communication is used by the Network monitor for trouble shooting.

14. *BITMAP AND ICON BASED GUI:*

A user friendly bitmap and icon based GUI is provided using Microsoft Visual C++. This plays an important role as it provides easy of use to a new operator.

4.4 WHY NETWORK ANALYSER ?

Some of the existing network monitoring systems in the market are :

1. LAN Paraoh form AZURE Tech.
2. Foundation Manager from Protools Inc.
3. LAN Decoder from TRITICOM

These do not support certain functions which could help the network administrator in making critical decisions to enhance the network performance.

ıe proposed network monitor will support the additional features like :

Intrasegment communication
RMON support
Protocol wise listing of information
Node wise/error wise listing of information
Application recognition
Play back option

4.5 FEATURES DESIRED IN THE FRONT END

The user interface for this project is done keeping in mind some basic principles in User Interface Design. They are :

- All applications running on a user's workstation should have a consistent interface design. Programs that deviate from the expected design will almost assuredly confuse the user even if the changes were intended for the users' benefit. Chances are also high that the user will not want to use the questionable software again.

- Users rely on rote memory ; they will remember seemingly complicated interface interaction techniques provided that the functions they perform are useful and are invoked frequently. There is a limit , however to how much users want to remember. It is important that essential or frequently used functions follow memorable patterns.

- Users, especially novices will probably not want to customise or alter their applications in any way. If they do, the available methods must be as easy and as painless as possible.

- The functionality should not be restricted to accommodate simplicity.

5 SYSTEM DESIGN AND IMPLEMENTATION

1 INTRODUCTION

The design for the system is a plan for a solution, such that the implementation the plan will satisfy the design objectives. The design process has two levels . the first level called system design, the components in the system and the way which the components communicate is taken care . The second level involves ne detailed design of the individual modules, which expands the system design to ontain more detailed description of the processing logic and data structures such nat the design is sufficiently complete for coding. This chapter produces the verall design of the system being developed, followed by the detailed design and mplementation details of each implemented module of **NetworkAnalyser.**

5.2 OVER ALL DESIGN OF NETWORK ANALYSER

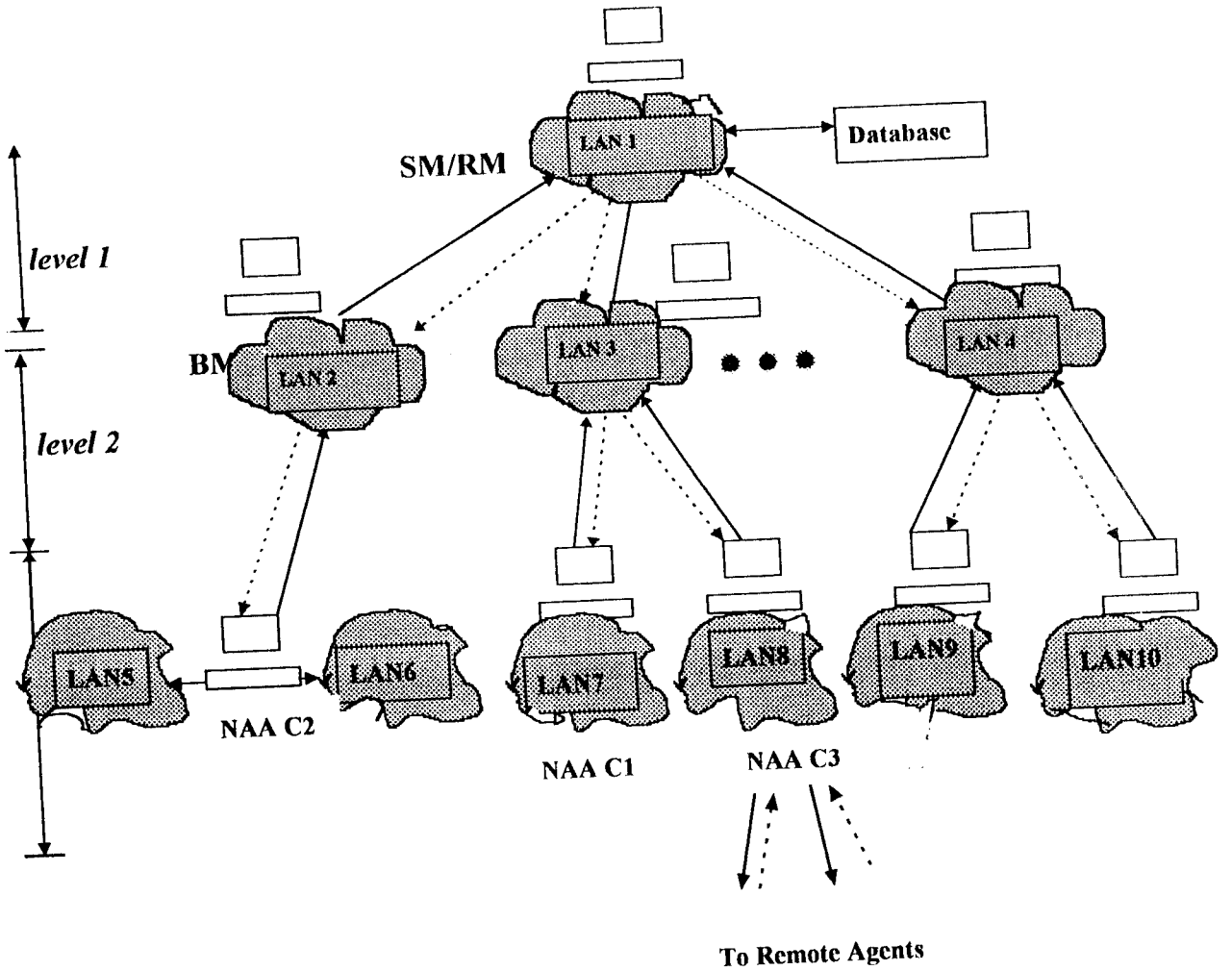
NetworkAnalyser comprises of the following components:

1. Back End
2. API modules
3. Front End

The physical components of NetworkAnalyser are :

1. Agents - where Network Analyser Back End is loaded
2. Branch Managers - contains API modules and user interface
3. SNMP Proxy agents - Switches for RMON
4. Super Manager / Remote Manager

2.1 DIAGRAMMATIC REPRESENTATION OF DESIGN



- Network Analyser Agent
  - Category C1 - Category 1(Host as Agent)
  - Category C2 - Category 2(Agent cum Bridge)
  - Category C3 - Category 3(Agent as Switch/Hub Remote Agent)
- Branch Manager

RM - Super Manager/Remote Manager

Data Flow

Control Flow

5.2.2 COMPONENT DETAILS :

THE AGENT:

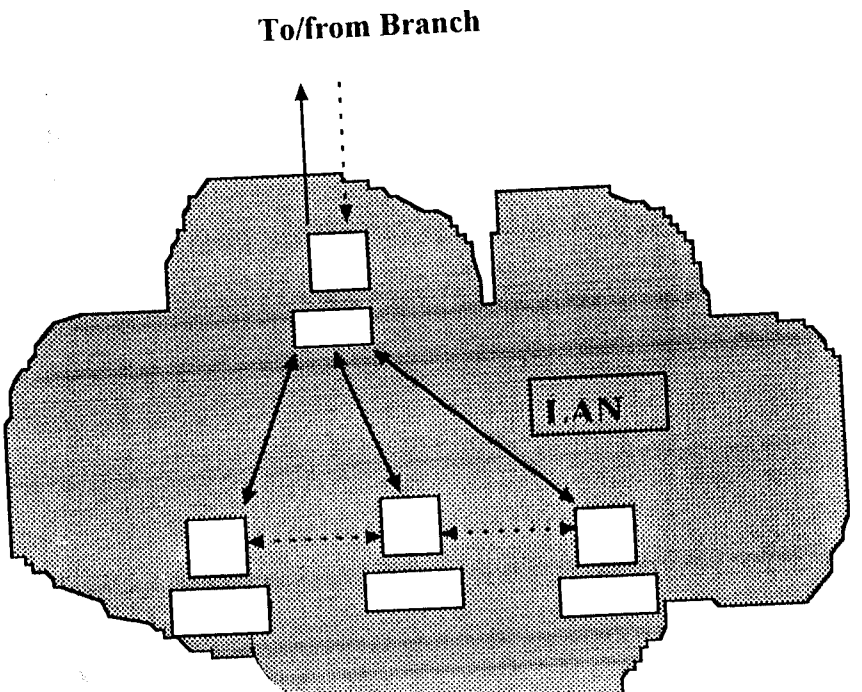
Each sub-network of interest has a host acting as an Agent. The Network Analyser is loaded in the Agents. Each agent monitors it's respective sub-network and updates the Network Information Base (NIB) which is present at every agent at regular time intervals of approximately four seconds which is the optimal time interval. The agent besides monitoring the network may carry on other activities. Depending the activities carried on by the agents, they are categorised into three types:

1. Single segment agents
2. Multiple segment agents
3. Switch / Hub agents

1. *SINGLE SEGMENT AGENTS:*

An agent which is a general host on the sub-network carrying on the normal activities of a node in the network is known as the Single Segment Agent (SSA) as shown below:

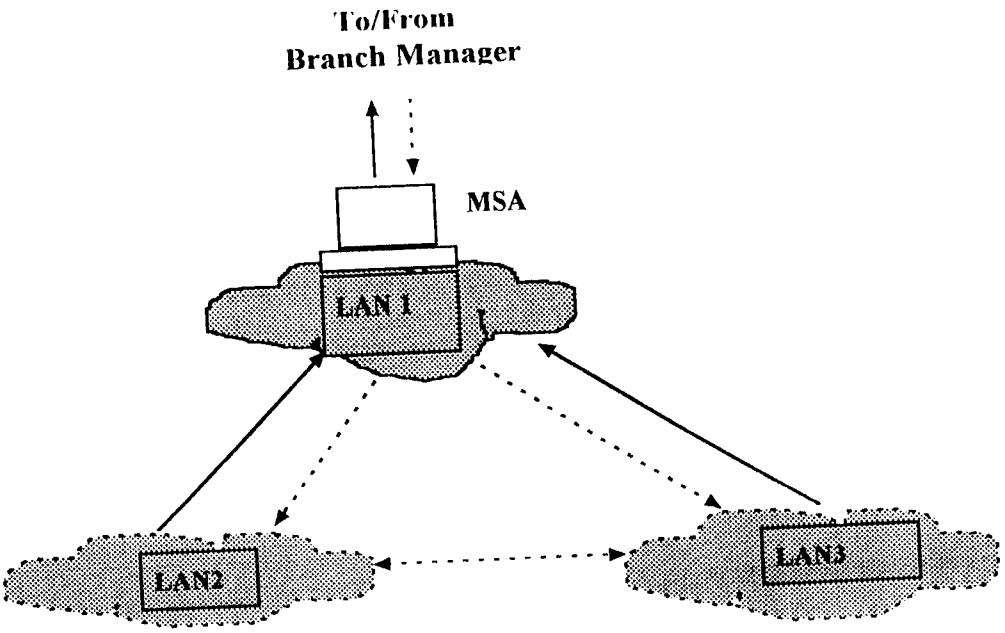
Diagrammatic representation of SSA:



2. *MULTIPLE SEGMENT AGENTS:*

An Agent which is itself a Branch Manager to a set of Agents is known as a Multiple Segment Agent (MSA). The MSA collects network details from it's agents and communicates to it's Branch Manager.

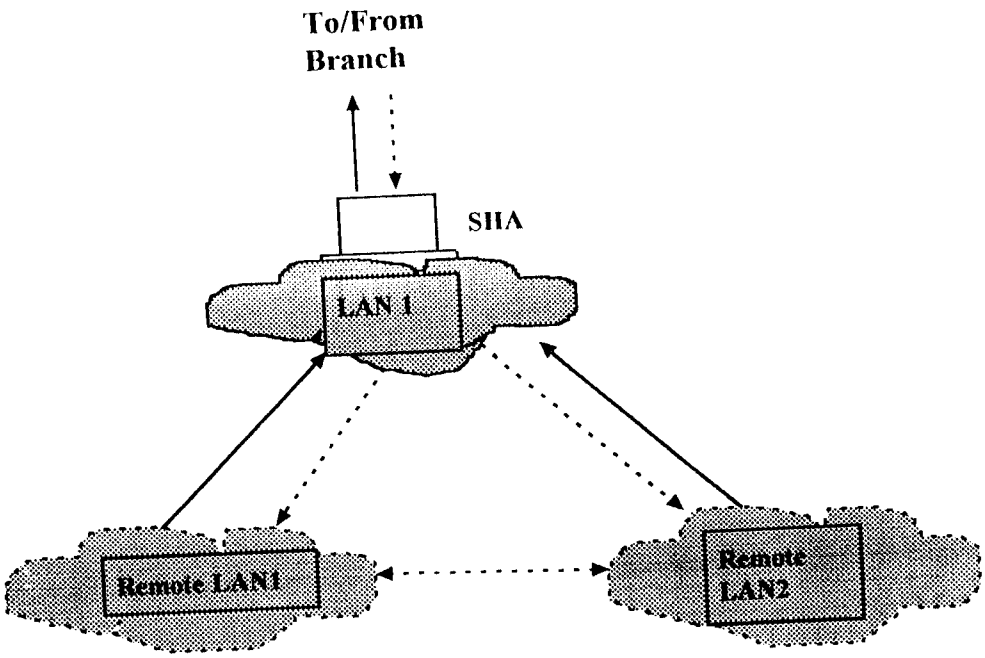
Diagrammatic representation of MSA:



3. *SWITCH OR HUB AGENTS (SHA):*

Switch or Hub Agents are those which monitor a remote network and convey network information of the same to the Branch Manager. An SHA should have the capability of protocol translation and synchronisation.

Diagrammatic representation of SHA:



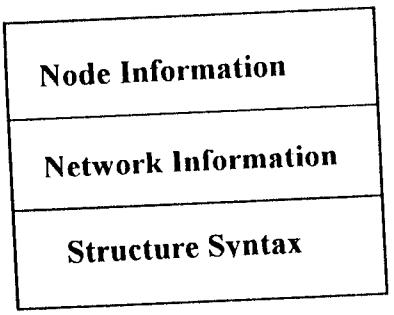
5.3 THE NETWORK INFORMATION BASE:

The ***NIB*** is a structure formed based on the rules known as Structure of Management Information (SMI) specification, the same specification used for the Management Information Base (MIB) of the SNMP. The NIB stores all the network information and is updated by the Agent at regular intervals of time (once in 4 seconds).

The NIB is organised to contain the following categories of information :

1. The data structure definition of NIB
2. Overall network information
3. Node wise information

Organisational Structure of NIB :



1. The data structure definition :

The data structure definition of the NIB contains the following variables and is basically a self referencing structure leading to a linked list representation.

5.3.1. ***TABLE OF NIB VARIABLES :***

VARIABLE	DATA TYPE	PURPOSE
Node_Number	WORD	Node Identifier
Node_Type □	Uchar	Type Identifier
Number_Of_Clients	Uchar	For server nodes only
Node_Address	Ushort	Physical addr of node
IP_Address	IP addr structure	Contains IP address
Node_name	Char array	Node identifier
Status □□	status structure	Status of node
Data_offset	WORD	Pointer to node area in memory
FirstClientOffset	WORD	Pointer to first client in case of Server
CommunicationNodeOffset	WORD	Offset of currently communicating node
Next	WORD	Pointer to the next node of same type

VARIABLE	DATA TYPE	PURPOSE
Activated_Time	DWORD	Time of activation of node
Time_Of LastUpdate	DWORD	Last updated time
No_Of_Packets_Out	DWORD	
No_Of_Packets_In	DWORD	
No_Of_ Bytes_Out	WORD	
No_Of_Bytes_In	WORD	
Max_Packet_Size	WORD	Size of biggest packet
Min_Packet_Size	WORD	Size of smallest packet

Packet Ranges :

These are variables to store the number of packets with sizes falling within a specific range.

Pkts0 - 63	DWORD
Pkts64 - 149	DWORD
Pkts150 - 249	DWORD
Pkts250 - 349	DWORD
Pkts350 - 449	DWORD
Pkts450 - 549	DWORD
Pkts550 - 749	DWORD
Pkts750 - 1049	DWORD
Pkts1050 - 1249	DWORD
Pkts1250 - 1517	DWORD
Pkts1518 - above	DWORD

BroadCast_ Packets	DWORD	(number of pkts)
MultiCast_ Packets	DWORD	(number of pkts)

VARIABLE	DATA TYPE	PURPOSE
ERROR VARIABLES : Each variable stores the number of packets with a specific type of error.		
Bad_CRC_ packets	DWORD	
FA_error_packets	DWORD	
FIFO_overrun_packets	DWORD	
Jabbers	DWORD	
Runts	DWORD	
Miscellaneous_Errors	DWORD	
Protocol	WORD	Indicates type of protocol used by node

- Node Type may be one of the following :
  1. Server
  2. Client
  3. General
  4. Existing new node
  5. Deleted node

- Status of a node may be one of the following :

  1. Repeater
  2. Router
  3. Bridge
  4. Gateway
  5. Active
  6. Hub
  7. Mail Server
  8. Print Server
  9. File Server
  10. Database Server
  11. Miscellaneous Server

2. Over all network information :

This portion of the memory contains the averages and totals of the various network parameters in terms of the entire network.

VARIABLE	DATA TYPE	PURPOSE
Activated_Time	DWORD	Time of activation of node
Time_Of LastUpdate	DWORD	Last updated time
No_Of_ Packets Out	DWORD	
No_Of_ Packets In	DWORD	
No_Of_ Bytes_Out	WORD	
No_Of_Bytes_In	WORD	
Max_Packet_Size	WORD	Size of biggest packet
Min_Packet_Size	WORD	Size of smallest packet

Packet Ranges : These are variables to store the number of packets with sizes falling within a specific range.

Pkts0 - 63	DWORD	
Pkts64 - 149	DWORD	
Pkts150 - 249	DWORD	
Pkts250 - 349	DWORD	
Pkts350 - 449	DWORD	
Pkts450 - 549	DWORD	
Pkts550 - 749	DWORD	
Pkts750 - 1049	DWORD	
Pkts1050 - 1249	DWORD	
Pkts1250 - 1517	DWORD	
Pkts1518 - above	DWORD	
BroadCast_ Packets	DWORD	(number of)
MultiCast_ Packets	DWORD	(number of)

VARIABLE	DATA TYPE	PURPOSE

Error Variables :Each variable stores the number of packets with as specific type of error.

Bad_CRC_ packets	DWORD
FA_error_packets	DWORD
FIFO_overrun_packets	DWORD
Jabbers	DWORD
Runts	DWORD
Miscellaneous_Errors	DWORD
Total_Error_Packets	DWORD

Protocols : Indicates number of nodes in the network using the particular protocol.

TCP	WORD
ARP	WORD
VINES	WORD
NetWareLite	WORD
Novell	WORD
NFS	WORD
3Com	WORD
NetBios	WORD
Miscellaneous	WORD

3. Node Wise Information :

A node may be one of the following types :

- Server
- Client
- General
- Remote
- New

Node information occupies the remaining of the memory space allocated for the NIB.

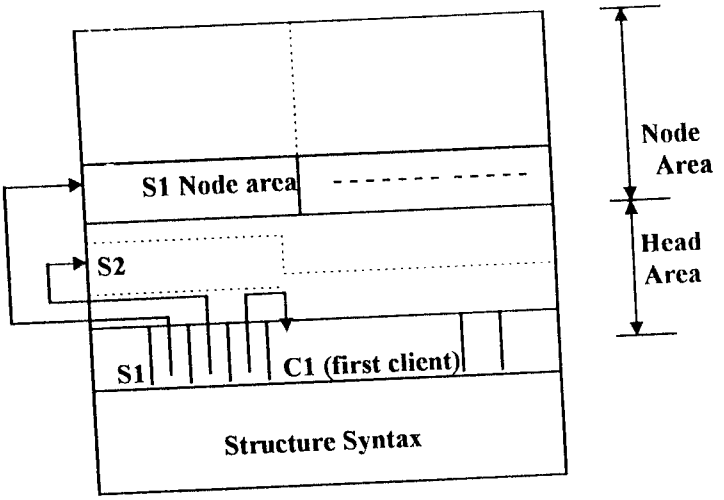
Every node is organised into two areas :

- Head Area
- Node Area

The static information of the node such as node name is placed in the Head Area in the memory and information subject to constant change such as number of packets into the node , is placed in the Node Area in the memory with pointers to link the head area and the node area . The head area also contains a pointer to the next node of it's kind and if it is a server, contains a pointer to the head area of it's first client. This type of organisation of the NIB enables easy browsing and extracting of information from it.

The following diagram represents the NIB organisation in the memory:

NIB ORGANISATION IN MEMORY :



Head area structure:

Node Num	Node Type	Node address	IP address	Node Name	status	Data offset	First Client offset	Comm. node offset	Next

Node area structure :

Act Time flag	Act Time	Time of last update	No. of Pkts.out	No.of Bytes out	No. Pkts. in	No. Bytes in	Max Packet Size

Min.Pkt Size	Pkts 0-63	Pkts 64-149	---- ---	Pkts 1518-above	No.Broadcast packets	No.Multicast packets

THE BRANCH MANAGER:

A Branch Manager controls a set of agents . There may be numerous Branch Managers depending on the size of the network . The User Interface is present at the Branch Manager. On request from the user for some network information about a particular node in a particular sub-network, the Branch Manager obtains the relevant information from the Network Information Base of the respective Agent using the API modules and displays the same to the User.

Communication between Branch Managers enables a BranchManager to get information about a particular node under the control of another Branch Manager.

THE SNMP PROXY AGENT:

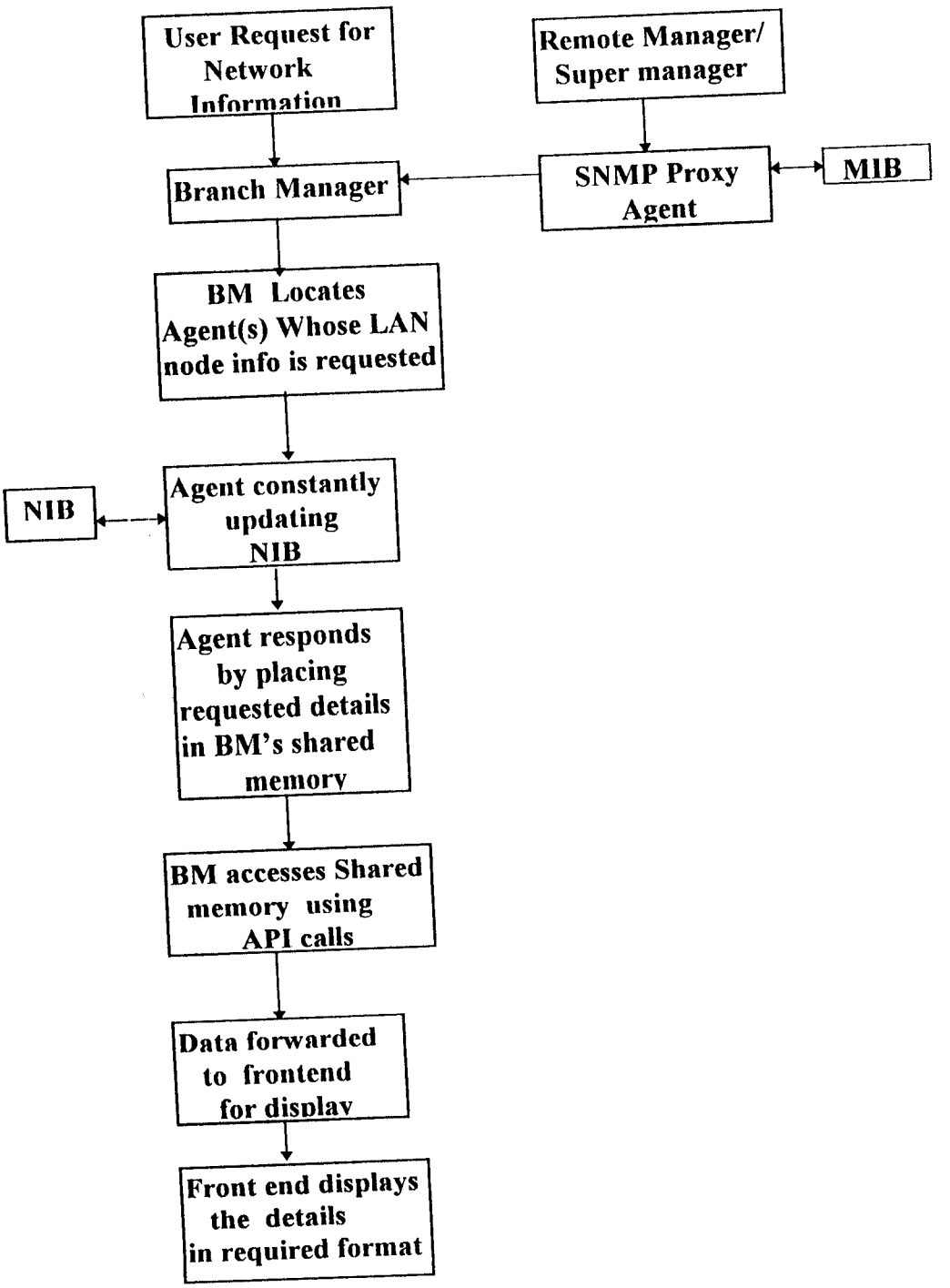
The SNMP Proxy Agent is an add on module to the Branch Manager that makes the branch manager an SNMP agent to other generic SNMP managers. Every Branch Manager is associated with a proxy agent. The Proxy Agent is mandatory for RMon (Remote Monitoring). The proxy agent receives data from the agents and stores the data in the RMON MIB and provides this information to the Super Manager on request.

THE REMOTE MANAGER:

Remote Manager is any node which is remote to the Network Analyser components and requires some network information about any of the nodes monitored by Network Analyser . The Remote Manager communicates with Network Analyser through the SNMP Proxy Agent. The Remote Manager could act as the SuperManager and control the entire Network Analyser set up.

5.4 FLOW OF COMMUNICATION AMONG COMPONENTS

On request for information from the user , the branch manager in turn requests the concerned agent for the information. The agent now places the NIB contents into the shared memory present at the Branch Manager. The Branch Manager forwards the required information to the front end which presents the information in the required format to the user.



5.5 DETAIL DESIGN OF IMPLEMENTED MODULES

This section deals with detailed description of the design of the following modules:

- API module
- Front End module for Network Information

5.5.1 DESIGN DESCRIPTION OF API MODULE

The API module, loaded in the Branch Manager acts like a middle man between the backend (present at the agents) and the front end (at the branch manager).
The user may request for information in one of the following modes :

- real mode – to view current information of the network which is being constantly updated at regular intervals of time.

- replay mode – a snapshot of the NIB at any desired instant is taken and stored in a back up file from which the user can get information current to that particular instant of recording.

REAL MODE OPERATION :

If real mode information of a particular network is requested , the Branch Manager fetches the NIB information from the corresponding agent and dumps into it's shared memory . This process is done in regular intervals so as to provide on line information to the user.

The shared memory is created using the concept of creating file mapping and then mapping the view of the NIB into the corresponding area committed in the Virtual Memory.

The SDK functions are used for implementing the shared memory concept :

- CreateFileMapping

This function creates a named or unnamed file mapping object for the specified file. It returns a handle which has full access to the new file mapping object.

CreateFileMapping object creates the potential for mapping a view file but does not map the file. The MapViewOfFile and MapViewOfFileEx functions are used for this purpose.

- MapViewOfFile

This function maps a view of a file into a process address space. Mapping a file makes the specified portion of the file visible in the address space of the calling process.

- OpenFileMapping

OpenFileMapping function is used to open a file mapping object.

- MapViewOfFileEx

This maps a view of a file into the address space of the calling process. This extended function allows the calling process to specify a suggested memory address for the mapping view.

- VirtualAlloc

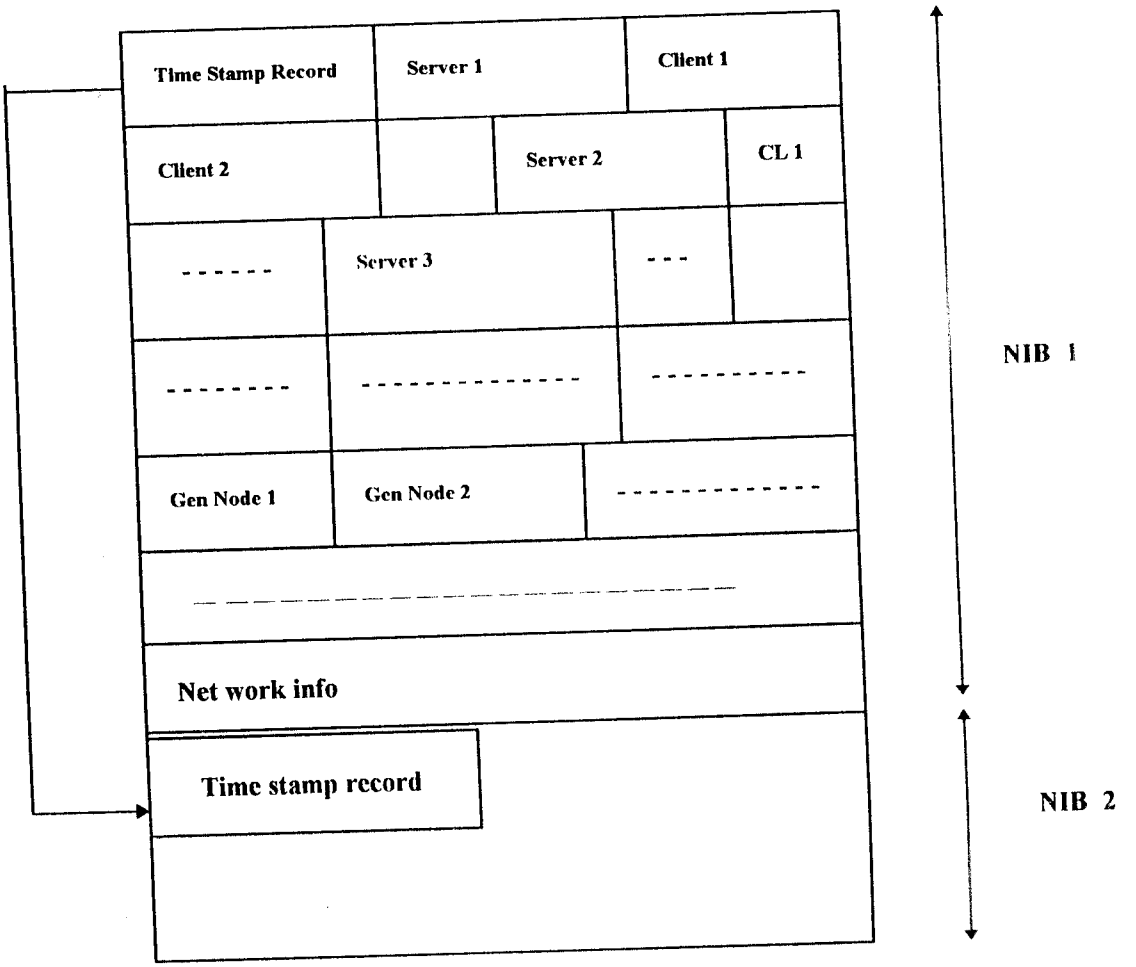
This function reserves or commits a region of pages in the virtual address space of the calling process. Memory allocated by this function is automatically initialised to zero.

REPLAY MODE OPERATION :

The replay mode allows the administrator to review the vital parameters of the network at any time for the past 30 minutes. This feature requires that the NIB has to be periodically dumped into a series of files. For example, if the NIB is dumped at 3 second intervals, for a replay period of 30 minutes, ten dump files are to be used. Using these files, the front end can display the required information.

The back up file is organised in a specific way which permits easy access of required information. The file structure is represented in the figure below :

File Organisation Diagram :



As shown in the above diagram , the file organisation of the NIB is as follows:

- each file contains the specified number of NIB snapshots taken at specified time intervals (usually 75 snapshots taken at 4 second intervals)
- each NIB snapshot begins with a Time Stamp Record which contains the following details :

Time Stamp Record structure:

The structure contains the following variables :

VARIABLE	DATATYPE
Day	integer
Month	integer
Year	integer
Hour	integer
Minutes	integer
Seconds	integer
NumberOfNodesInNIB	integer
NextNibOffset	long

- The Time Stamp Record is followed by each server node(with head area and node area information) and it's list of client nodes (with head and node area information).
- The next part of the file contains the general nodes also with head and node area information.
- Each NIB snapshot is concluded with the entire-network information.
- Each Time Stamp Record contains the offset of the next Time Stamp Record.

5.5.2 FUNCTIONING OF THE API MODULE :

The API module which is basically an interface handler is created as a regular Dynamic Link Library (DLL) with MFC Library support.

The DLL contains the following functions which contain code to access the required information from the NIB.

S.No.	Function Name	Service Provided
1	GetClientsForServer	gives complete details of clients of specified server
2	GetServerForClient	gives complete server details of specified client
3	GetNodeInfo	gives full information about specified node
4	GetNodesForProtocol	lists the nodes which use the specified protocol
5	GetUtilisationOfNetwork	calculates the utilisation of the network at that particular instant of time
6	GetUtilisationOfNode	calculates the utilisation of a specific node at that particular instant of time
7	GetTrafficOfNode	gives the traffic generated by a specific node
8	GetTrafficOfNW	gives the traffic generated by the network at that instance of time

S.No.	Function Name	Service Provided
9	GetTrafficUtilForNW	gives the traffic or utilisation for a specific node
10	GetNWParameter	gives the value corresponding to any parameter in the network
11	GetNodeParameter	gives the value corresponding to any parameter for any specific node
12	GetErrorForNW	returns the number of errors that occurred in the network for any specific type of error
13	GetErrorForNode	gives the number of errors generated by a node for any specific type of error
14	GetProtocolForNode	gives the protocol used by a specific node
15	GetNodeList	gives full details about all the nodes in the network
16	GetServerList	gives full details about the servers in the network.
17	GetTraffUtilList	gives a list containing the node name and it's corresponding traffic or utilisation
18	GetErrorList	gives the node name and the number of errors generated by the node for the specified error type

S.No.	Function Name	Service Provided
19	GetPackRange	gives the node name and it's corresponding parameter value for all nodes in the network
20	SortValue	returns a list whose nodes are sorted in a specific order based on a specified parameter
21	FindError	gives the nature of error

These functions kept to the minimum to minimise overheads, not compromising on the functionality or performance at the same time have been designed optimally. With this DLL containing the core functions, the front end can access any network information very easily.

DLL DETAILS:

1. GetClientsForServer

The function gives full information about the clients connected to a particular server.

NodeInfoPtr far * **GetClientsForServer** (char far * *ServerName*)

Parameters:

ServerName - The name of the server whose clients' details is required.

Return Value :

If the function succeeds it returns a pointer to the first node in the linked list.
In case of an Error NULL is returned . Use FindError ( ) to get more information about the error.

2. GetServerForClient

The function gives full information about server to which the specified node is a client.

NodeInfoPtr far * **GetServerForClient** (char far * *ClientName*)

Parameters :

ClientName - The name of the client whose server's details are required.

Return Value :

If the function succeeds it returns a pointer to the first node in the linked list .
In case of an Error NULL is returned .Use FindError ( ) to get more information about the error.

3. GetNodeInfo

The function gives full information about any specific node.

NodeInfoPtr far * **GetNodeInfo** (char far * *NodeName*)

Parameters :

NodeName - The name of the node whose details are required.

Return Value :

If the function succeeds it returns a pointer to the first node in the linked list .
In case of an Error NULL is returned .Use FindError ( ) to get more information about the error.

4. GetNodesForProtocol

The function gives the list of nodes in the network which use a specific protocol.

NodeInfoPtr far * **GetNodesForProtocol** (WORD *Protocol*)

Parameters :

Protocol - A predefined constant indicating the protocol.It can be one of the following constants.

TCP	ARP	Vines
NetwareLite	Novell	NFS
P3Com	NetBios	

Return Value :

If the function succeeds it returns a pointer to the first node in the linked list .

In case of an Error NULL is returned.Use FindError ( ) to get more information about the error.

5. GetUtilisationOfNW

The function calculates the utilisation factor of the network at that particular instance of time.
This function has no parameters.

float FAR PASCAL **GetUtilisationOfNW** (void)

Return Value :

If the function succeeds it returns a value denoting the Utilisation factor of the network.
In case of an Error -1 is returned.Use FindError( ) to get more information about the error.

6. GetUtilisationOfNode

The function calculates the utilisation factor of a specific node at that particular instance of time.

float FAR PASCAL **GetUtilisationOfNode** (char far * *NodeName*)

Parameters :

NodeName - The name of the node whose utilisation factor is required.

Return Value :

If the function succeeds it returns a value denoting the Utilisation factor of the node NodeName.
In case of an Error -1 is returned.Use FindError ( ) to get more information about the error.

7. GetTrafficOfNode

The function gives the traffic generated by a specific node.

float FAR PASCAL **GetTrafficOfNode** (char far * *NodeName*)

Parameters :

NodeName - The name of the node whose traffic is required.

Return Value :

If the function succeeds it returns a value denoting the traffic generated by the node NodeName.
In case of an Error -1 is returned.Use FindError( ) to get more information about the error.

8. GetTrafficOfNW

The function gives the trafffic generated by the network at that instance of time.
The function has no parameters.

float FAR PASCAL **GetTrafficOfNW** (void)

Return Value :

If the function succeeds it returns a value denoting the traffic generated by the network.
In case of an Error -1 is returned.Use FindError ( ) to get more information about the error.

9. GetTraffUtilForNode

The function gives the traffic or utilisation for a specific node.

float FAR PASCAL **GetTraffUtilForNode** (char far * *NodeName*,WORD *TrafUtil*)

Parameters :

NodeName - The name of the node whose traffic or utilisation is required.
TrafUtil - Constant indicating the required parameter.It can be either TRAFFIC or UTILISATION

Return Value :

If the function succeeds it returns a value denoting the traffic or utilisation factor generated by the node NodeName.
In case of an Error -1 is returned . Use FindError( ) to get more information about the error.

10. GetNWParameter

The function gives the value corresponding to any parameter in the network.

LONG FAR PASCAL **GetNWParameter** (WORD *NWParameter*)

Parameters :

NwParameter - A predefined constant indicating the Network Parameter.
It can be one of the following.

PKTS0_64
PKTS64_150
PKTS150_250
PKTS250_350

PKTS350_450
PKTS550_750
PKTS1050_1250
PKTS1518_ABOVE
BROADCAST
CRCPKTS
FIFOOVERRUN
RUNTS
TOTALERRPKTS
TCP
Vines
Novell
Com
MISCPROTOCOL
COLLISION
DROPPED
NOOFPKTSIN
MINPKTSIZE

PKTS450_550
PKTS750_1050
PKTS1250_1518

MULTICAST
FAERROR
JABBER
MISCERROR

ARP
NetwareLite
NFS
NETBIOS
CHAACQTIME
DEFFERED
NOOFNODES
MAXPKTSIZE

Return Value:

If the function succeeds it returns a value corresponding to the specified NWParameter for the network.
In case of an Error -1 is returned.Use FindError ( ) to get more information about the error.

11. GetNodeParameter

The function gives the value corresponding to any parameter for any specific node.

LONG FAR PASCAL **GetNodeParameter** (char far * *NodeName* ,WOR
NodeParameter)

Parameters :

NodeName - The name of the node whose Parameter value is required.
NodeParameter - A predefined constant indicating the Network Parameter.

Return Value :

If the function succeeds it returns a value corresponding to the specified NWParameter for the node *NodeName*.
In case of an Error -1 is returned.Use FindError( ) to get more information about the error.

12. GetErrorForNW

The function gives the number of errors that occurred in the network for any specific type of error.

LONG FAR PASCAL **GetErrorForNW** (WORD ErrorType)

Parameters :

ErrorType - A predefined constant indicating the type of error.
It can be one of the following

CRCPKTS	FAERROR
FIFOOVERRUN	JABBER
RUNTS	MISCERROR
ERRORPKTS	

Return Value :

If the function succeeds it returns the number of errors of type ErrorType that occurred in the network.
In case of an Error -1 is returned.Use FindError( ) to get more information about the error.

13. GetErrorForNode

The function gives the number of errors generated by a node for any specific type of error.

LONG FAR PASCAL **GetErrorForNode** (char far * *NodeName*,WORD *ErrorType*);

Parameters :

NodeName - The name of the node whose error details are required.
Error Type - A predefined constant indicating the type of error. Error Type should be one of those as specified in 12.

Return Value :
If the function succeeds , it returns the number of Errors of type ErrorType for the node NodeName.
In case of an Error -1 is returned . Use FindError ( ) to get more information about the error.

14. GetProtocolForNode

The function gives the protocol used by a specific node.

LONG FAR PASCAL **GetProtocolForNode** (char far * *NodeName*)

Parameters :

NodeName - The name of the node whose protocol detail is required.

Return Value :

If the function succeeds , it returns the Protocol used by the node NodeName.This value is a predefined constant.
In case of an Error -1 is returned . Use FindError( ) to get more information about the error.

15. GetNodeList

The function gives the full details about all the nodes in the network. The function has no parameters.

Nodeinfo far* FAR PASCAL **GetNodeList** ( )

Return Value :

If the function succeeds , it returns a pointer to the first node in the linked list.
In case of an Error NULL is returned.Use FindError ( ) to get more information about the error.

16. GetServerList

The function gives full details about the servers in the network.The function has no parameters.

NodeInfoPtr far* FAR PASCAL **GetServerList** ( )

Return Value :

If the function succeeds , it returns a pointer to the first node in the linked list.
In case of an Error NULL is returned.Use FindError ( ) to get more information about the error.

17. GetTraffUtilList

The function gives a list containing the nodename and its corresponding Traffic or Utilisation.

SortInform far* FAR PASCAL **GetTraffUtilList** (WORD TrafUtil)

Parameters :

TrafUtil - Constant indicating the required parameter.It can be either TRAFFIC or UTILISATION

Return Value :

If the function succeeds , it returns a pointer to the first node in the linked list.
In case of an Error NULL is returned.Use FindError ( ) to get more information about the error.

18. GetErrorList

The function gives the nodename and the number of errors generated by the node for the specified errortype.

SortInform far* FAR PASCAL **GetErrorList** (WORD ErrorType)

Parameter :

ErrorType - A predefined constant indicating the type of error. ErrorType should be one of those as specified in 12.

Return Value :

If the function succeeds , it returns a pointer to the first node in the linked list.
In case of an Error NULL is returned.Use FindError ( ) to get more information about the error.

19. GetPackRange

The function gives the nodename and its corresponding parameter value for all nodes in the network.

SortInform far* FAR PASCAL **GetPackRange** (WORD *Parameter*)

Parameters :

Parameter - A predefined constant indicating the parameter whose value is to be retrieved.

Return Value :

If the function succeeds , it returns a pointer to the first node in the linked list.
In case of an Error NULL is returned.Use FindError ( ) to get more information about the error.

20. SortValue

The function returns a list whose nodes will be sorted in a specific order

SortInform far* FAR PASCAL **SortValue** (WORD Parameter,WORD Value,WORD Order);

Parameters :

Parameter - A predefined constant.It can be one of the following.
TRAFFIC UTILISATION PACKRANGE ERRTYPE

Value - value of node for specified parameter

Order - A predefined constant specifying the sorting order.It can be ASCENDING or DESCENDING.

21. FindError

The function gives the nature of error .The function has no parameters.

int FAR PASCAL **FindError** (void)

Return Value :

The value returned will be one of the following .

ERROR_NODE_NOT_SERVER
ERROR_NODE_NOT_FOUND
ERROR_NO_NODES
ERROR_NODE_NOT_CLIENT
ERROR_INVALID_PROTOCOL
ERROR_INVALID_PARAM
ERROR_CLIENT_NOT_FOUND

5.5.3 DESIGN DESCRIPTION OF INFOANALYSER , FRONT END MODULE FOR NETWORK INFORMATION

The front end module for Network Information is one of the important modules of the front end .It consists of the following sub-modules :

1. Server Information
2. Node Information
3. Error Information

1. *Server Information*

Various options are provided to view required information,as follows :

OPTIONS :

a.Client Information

Client information is displayed under two divisions:

- Client Details - Clients of the Server are listed and detail information of any of the clients may be displayed.
- Sorted List of Clients - Based on the value of a parameter, the clients are displayed in ascending or descending order.

b. Server Comparison

Servers are compared based on the following categories:

- Packet range wise comparison
- Error wise comparison
- Parameter wise comparison

c. **Server Details**

The vital static and dynamic information of the Server stored in the head area and node area in the NIB is displayed.

2. ***Node Information***

Node information is presented under two categories :

a. **Node Comparison**

- Error wise comparison
- Packet range wise comparison
- Parameter wise comparison

b. Node Details

Details of node as stored in the NIB are displayed.

3.***Error Information***

Error information is presented in three categories :

a. Error percent of each error type

The error percent of each error such as CRC %, RUNT % etc. in the network is displayed

b. Error wise listing of nodes

The values of a particular error in each node in the network are listed.

c. Total Error Percent

The error percent of each type of error and the total error percent in a specific node is presented

4. *Parameter Information*

Parameter details is displayed in all possible angles so as to help the administrator judge the network based on the parameter values. Parameter information is presented as follows:

a. Parameter wise listing of nodes

For any particular parameter, the node values are listed

b. Parameter wise comparison

Nodes are compared based on their values for a particular parameter. Nodes are listed in descending order of values in Parameter Maxima and in ascending order in Parameter Minima.

c. Parameter Details

The parameter values are displayed

- Node wise or
- in the Network as a whole

Thus by presenting crucial information in a user palatable form, the front end InfoAnalyser of NetWork Analyser achieves it's goal.

Coding and Testing

6 CODING AND TESTING

6.1 INTRODUCTION

The primary goal of the coding phase is to translate the given detail design to source code in a given programming language such that the code is simple, easy to test and easy to understand and modify. The programming style followed in the coding phase of the product development has been discussed in detail in the following section. In order to ensure that the system works satisfactorily and does not produce erroneous results, the system being developed has been tested thoroughly and the testing methods have been explained in detail in this chapter.

6.2 PROGRAMMING STYLE

The key aspect kept in mind while coding has been "writing optimal but decipherable code". Both these features of code are essential; optimal so that the desired functionality is achieved with minimum code , not compromising on the clarity at the same time.

The company has laid certain coding standards such as :

- Standards for Naming Variables
- Standards for Structure definitions
- Indentation Standards - TAB space setting
- Loops, Conditional statement Standards
- Function Naming Standards

etc.

Following those standards ensures optimal code which can be understood by any user familiar with the standards.

6.3 TESTING

Testing is an important step in completion of a project as it aims at perfection of the performance of the software. The API module and the front end have been tested using a back end simulator which generates various values for the NIB variables and checks the functioning of the modules. In replay mode , the modules have been testing with back up files taken at various instances thereby ensuring the generality of the modules.

Suggestions and Enhancements

7 SUGGESTIONS AND ENHANCEMENTS

The **Network Analyser** is in itself a self contained and complete tool that provides all the functionalities required by the network administrator to efficiently monitor and manage the network. But there are some extensions possible to the project which will increase the generality and functionality of the Network Analyser as follows.

- **Support for token ring :**

The first phase development of Network Analyser supports only Ethernet. Token ring, FDDI etc. are expected to be supported in the phases to come. This will provide the Network Analyser complete device independence and will permit the application of this network monitoring system to all underlying technologies, thus enhancing it's versatility.

- **Artificial Intelligence Engine :**

The AI concept could be applied extensively in the area of error detection and analysis. When errors occur , the AI engine could provide logical reasons for the same to the network administrator; inferences could also be made from previous similar 'experiences'. Implementation of this concept requires a full fledged knowledge based engine which again is planned for implementation in the phases to follow.

8 BIBLIOGRAPHY

1. **William Stallings** - "SNMP,SNMPv2,CMIP, The Practical Guide to Network Management Standards",MacMillan Publishing Company.

2. **Andrew S. Tanenbaum** - 'Computer Networks', 1993, Prentice Hall of India.

3. **Douglas Comer** - 'InterNetworking with TCP/IP-vol 1', third edition, Prentice Hall of India

4. **Ralph Davis** - "Windows NT Network Programming",Addison Wesley Publishing Company,1994.

5. **Helen Custer** - "Inside Windows NT", Microsoft Press, 1993

6. **David J. Kruglinski** - 'Inside Visual C++', 1993, MicroSoft series

7. **Steve Holzner** - 'Heavy Metal Visual C++ Programming', 1994, Pustak Mahal.

9 APPENDICES

9.1 NETWORKS - AN OVERVIEW

An interconnection of large number of autonomoust computers is called a Computer Network . Need for information,resource sharing , load sharing and high reliability lead to computer networks. Depending on the area in which the network spans , the computer networks are classifed into three types namely, Local Area Networks (LAN), Metropolitan Area Networks (MAN) and Wide Area Networks (WAN). A LAN spans over an area of 2 kilometres with very high data transfer rate and can be owned by a single organisation. A MAN covers an entire city and a WAN is an interconnection of LANs across the world. Connection of two or more networks which provides interoperability is called internetworking.

TERMINOLOGIES :

- **Node** - A device connected in the network is called a node. A node can be either be a computer or any other device.

- **Server** - A server is a specialised high performance computer on the network that provides specialised services to other nodes on the network . Typical servers include file servers , print servers, mail servers, database servers, etc.

- **Client** -A node that get services from a server is termed as a client of that server.

- **Peer - to - Peer Communication** -Each workstation is a client and if the chooses, it can also become a server by sharing it's local resources

with it's other clients.

- **Client - Server Communication** - In this model , servers provide services on request from their clients.

- **Network Interface Card** - Each node is connected to the actual network hardware by a Network Interface Card (NIC). The NIC in each node of Ethernet contains among other things, a *Erasable Programmable Read Only Memory (EPROM)* containing a unique number (which is the address of the node) , a 256K Circular Buffer into which frames are captured , a 8390 chip that controls the operation of NIC and Circular Buffer.

- **Network Topology** - The way in which the nodes in a network are connected using a physical media is known as Network Topology.

9.2 DYNAMIC LINK LIBRARIES - AN INSIGHT

A DLL is a file on disk consisting of global data, compiled functions and resources that becomes part of the user's process. It is compiled to load at a preferred base address and if there is no conflict with other DLLs, the file gets mapped to the same virtual address in the user's process. The DLL has various 'exported' functions and the client program imports those functions. Windows matches up the imports and exports when it loads the DLL.

A DLL contains a table of exported functions. These functions are identified to the outside world by their symbolic names and by integers called ordinal numbers. The function table also contains the addresses of the functions within the DLL. When the client program first loads the DLL, it does not know the address of the function it needs to call, but it does know the symbols or ordinals. The dynamic linking process then builds a table that connects the client's call to the function addresses in the DLL.

Even if the DLL is edited and rebuilt , the client program need not be rebuilt unless the function names or parameters have been changed.

9.3 RMON - AN INSIGHT

Network monitors are devices that traditionally have been employed to study the traffic on a network as a whole For the purposes of network management in an internetworked environment, there would typically need to be one monitor per subnetwork. For effective network management, these monitors need to communicate with a central network management station. In this context, they are referred to as **Remote Monitors.**

RMON GOALS :

The RMON is primarily a definition of an MIB. The effect however is to define standard network monitoring functions and interfaces between SNMP - based management consoles and remote monitors. The following are the design goals of RMON :

- **Off-line Operation :**

The monitor should collect fault, performance and configuration information continuously , even if it is not being polled by a network manager. The monitor simply continues to accumulate statistic that may be retrieved by the manager at a later time.

- **Pre-emptive Monitoring :**

If the monitor has sufficient resources,and if the practice is not considered too disruptive, the monitor can continuously run diagnostics and log network performance.

- **Problem Detection and Reporting :**

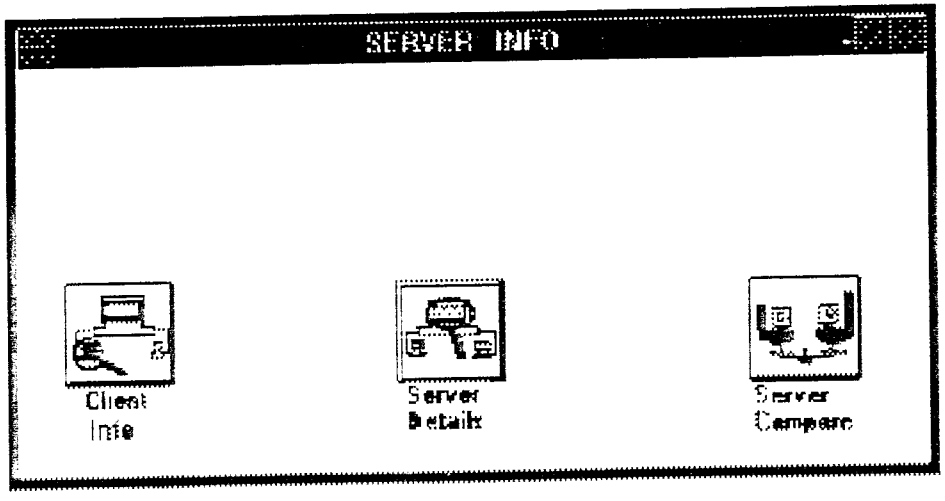
The monitor can passively recognise certain error conditions and other conditions such as a congestion, on the basis of the traffic that it observes. The monitor can be configured to continuously check for such condition.

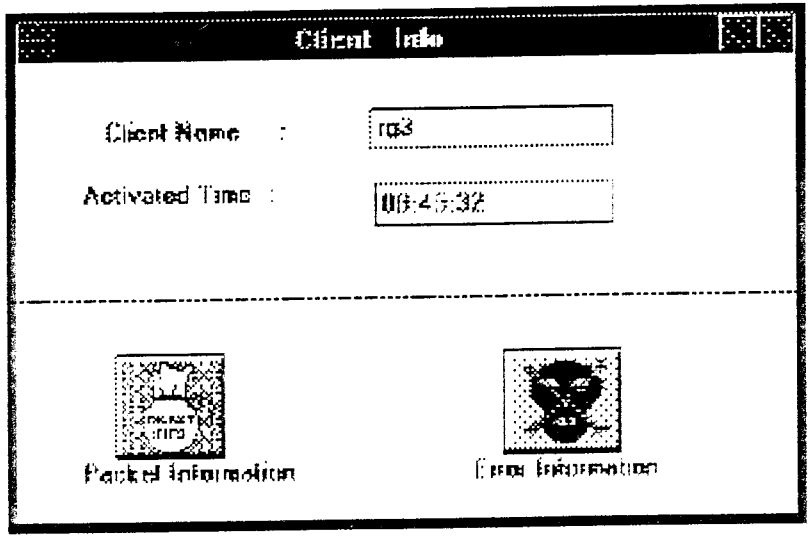
When one of these conditions occur, the monitor can log the condition and attempt to notify the management station.

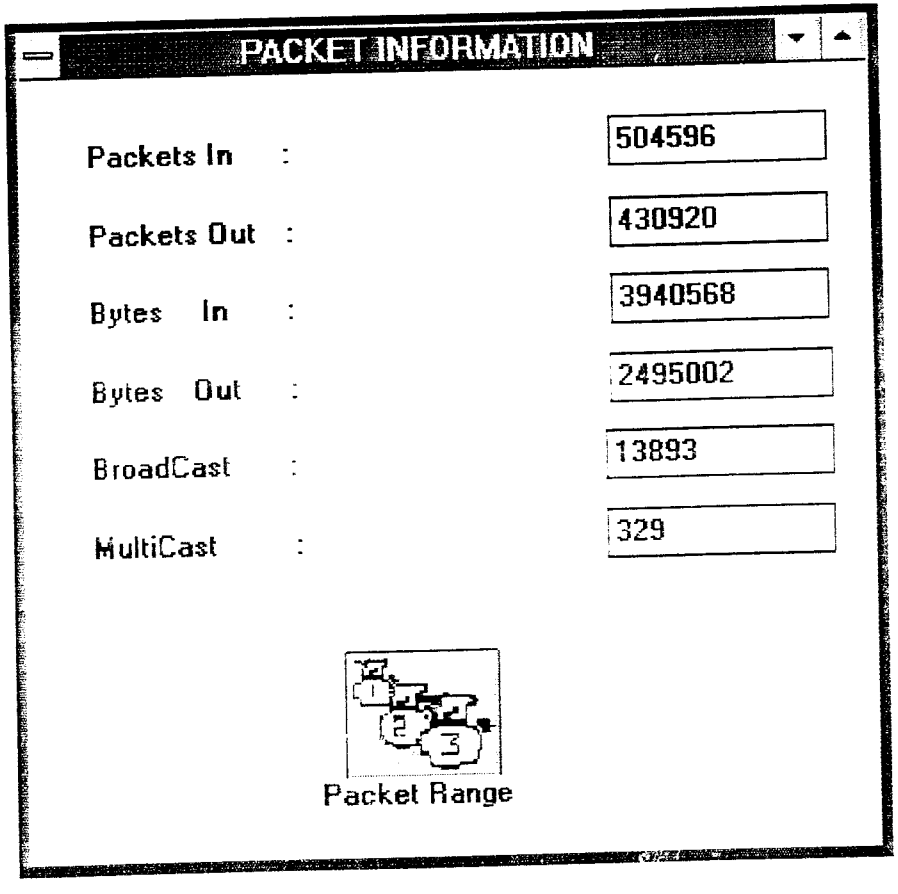
- **Value Added Data :**
  The network monitor can perform analysis specific to the data collected on it's subnetwork.

- **Multiple Managers :**
  An internetworking configuration may have more than one management station in order to achieve reliability , perform different functions and provide management capability to different units within an organisation.

9.4 SCREEN LAYOUTS

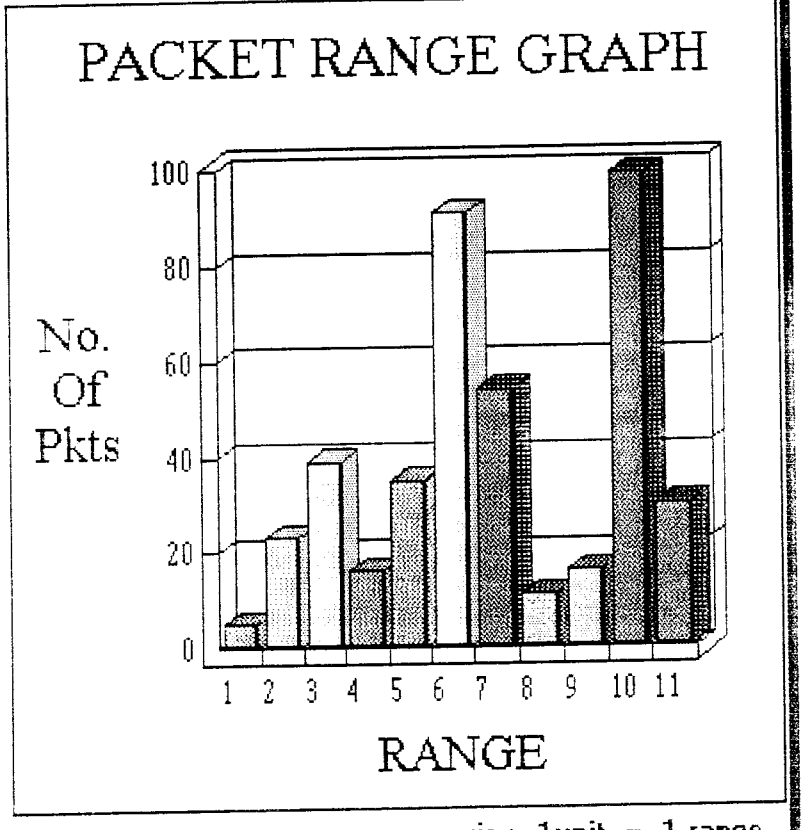

SERVER INFO
Client
Info
Server
Details
Server
Compare


Client Info
Client Name :
rq3
Activated Time :
Packet Information
Error Information

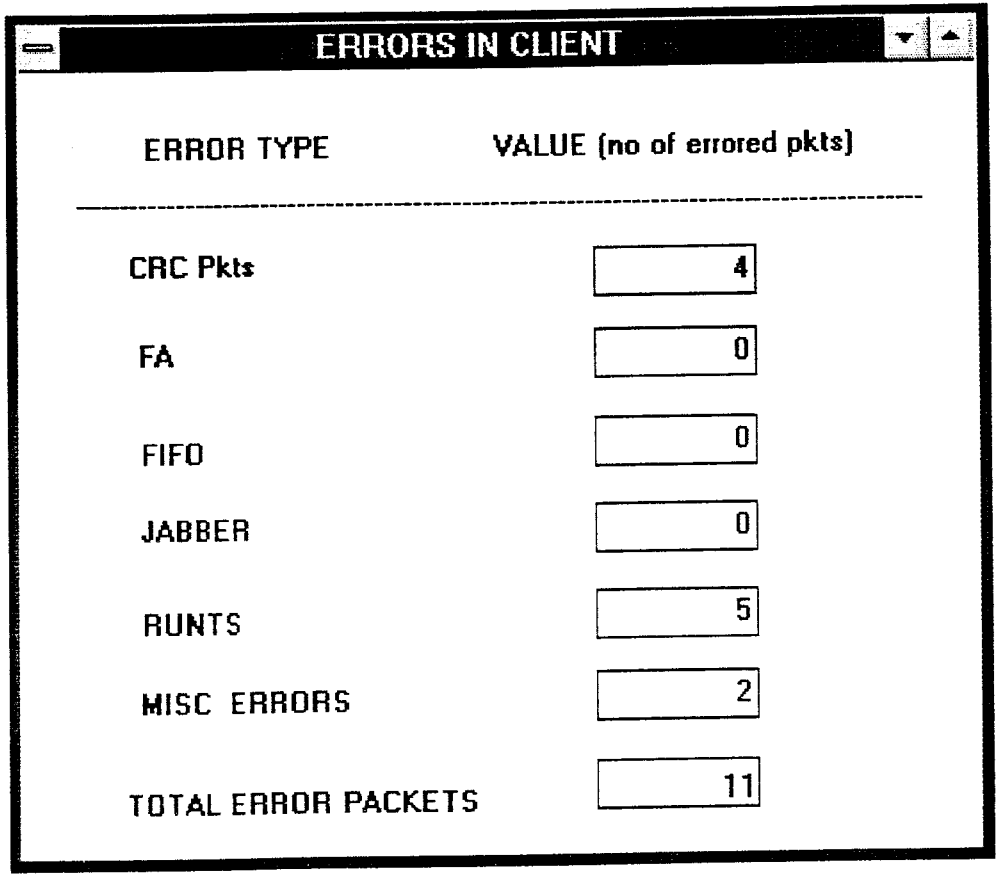

PACKET INFORMATION
Packets In :
504596
Packets Out :
430920
Bytes In :
3940568
Bytes Out :
2495002
BroadCast :
13893
MultiCast :
329
Packet Range

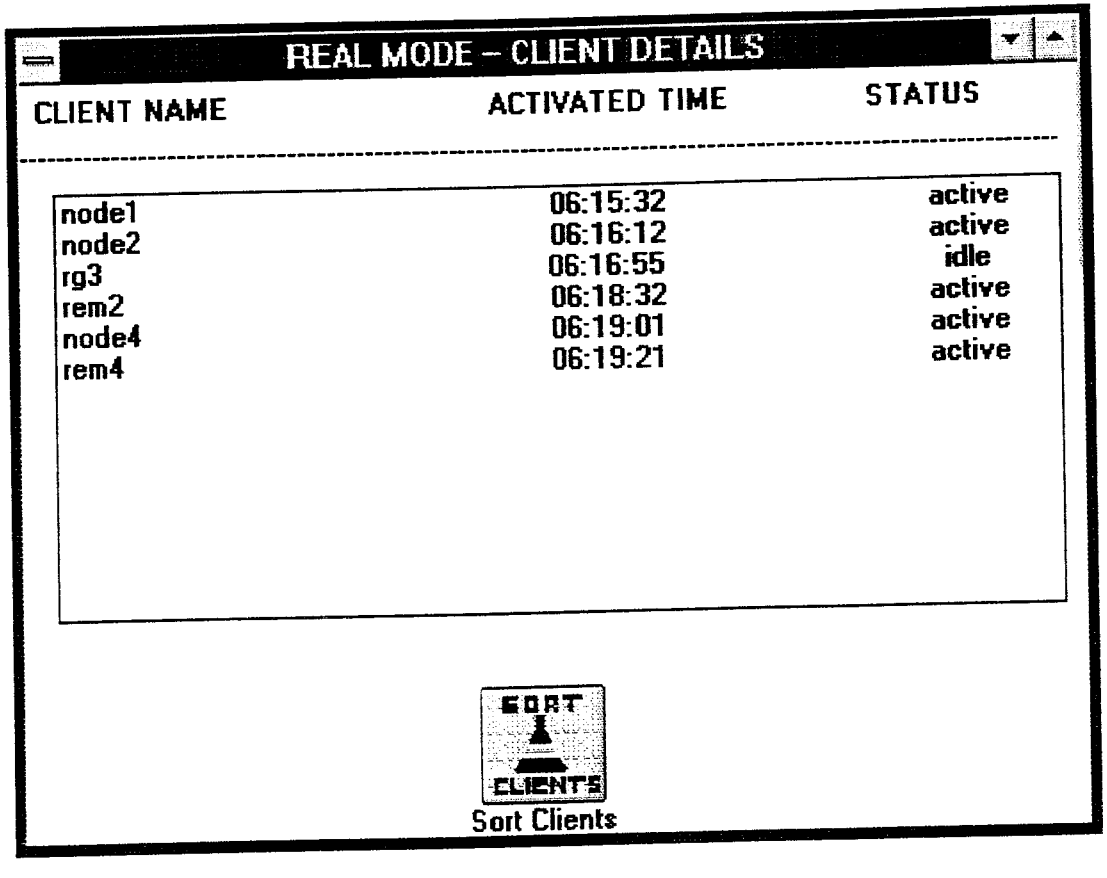
PACKET RANGE INFORMATION

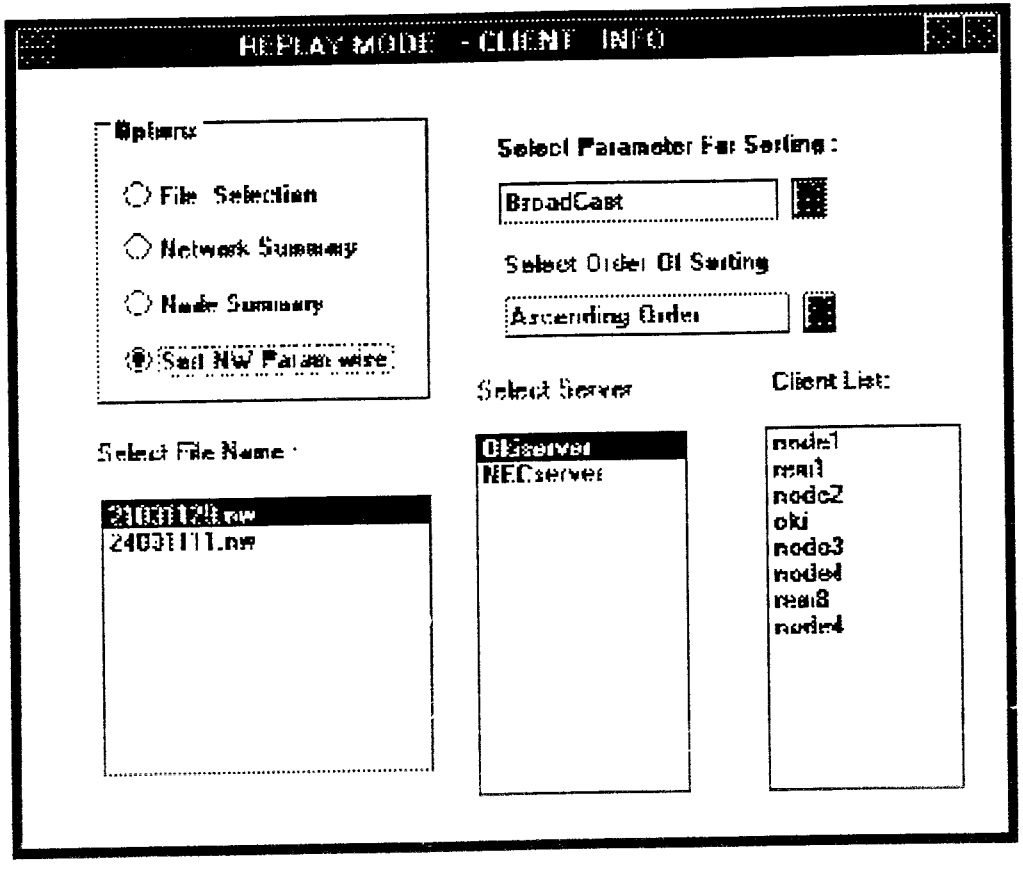
RANGE	VALUE
0 - 63	583
64 - 149	2129
150 - 249	3942
250 - 349	1768
350 - 449	3672
450 - 549	9130
550 - 749	5632
750 - 1049	1350
1050 - 1249	1532
1250 - 1518	9856
1518 - ABOVE	2854

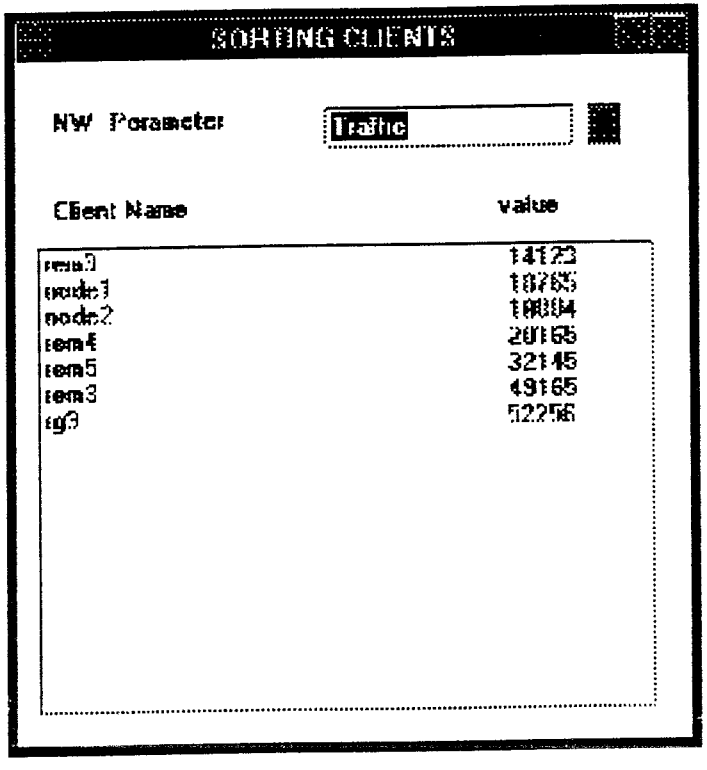


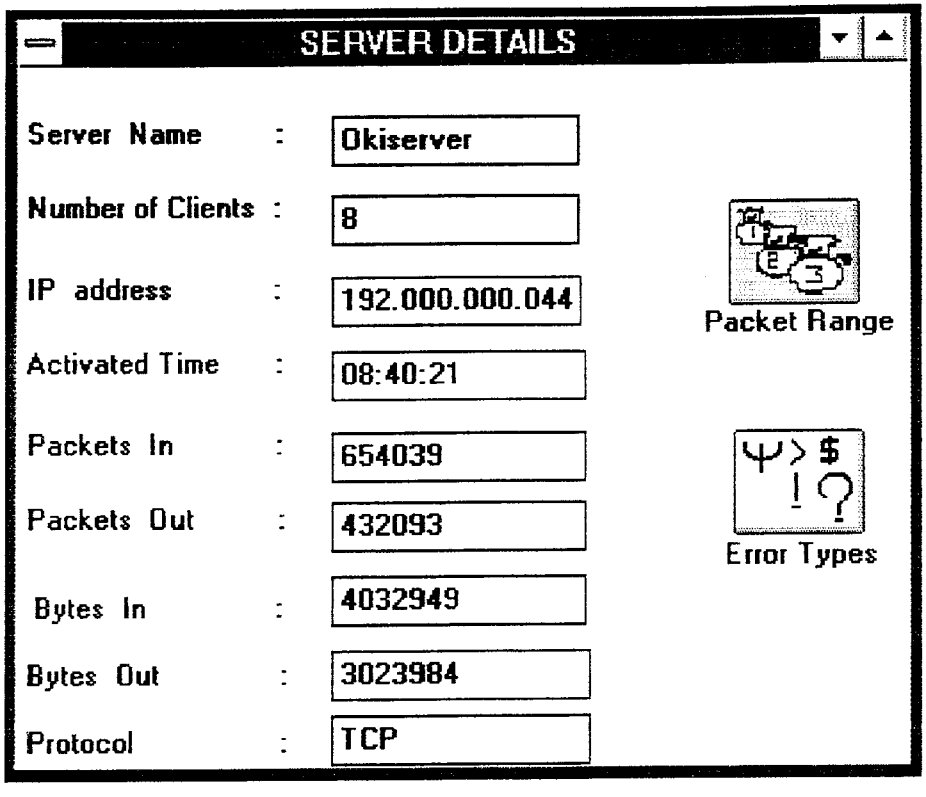
x axis : 1unit = 1 range
y axis : 1 unit = 100 pckts

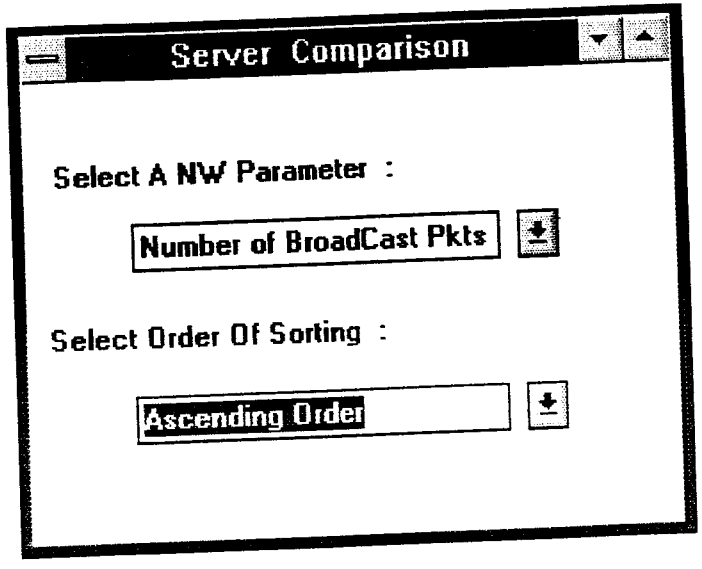

ERRORS IN CLIENT
ERROR TYPE
VALUE (no of errored pkts)
CRC Pkts
4
FA
0
FIFO
0
JABBER
0
RUNTS
5
MISC ERRORS
2
TOTAL ERROR PACKETS
11

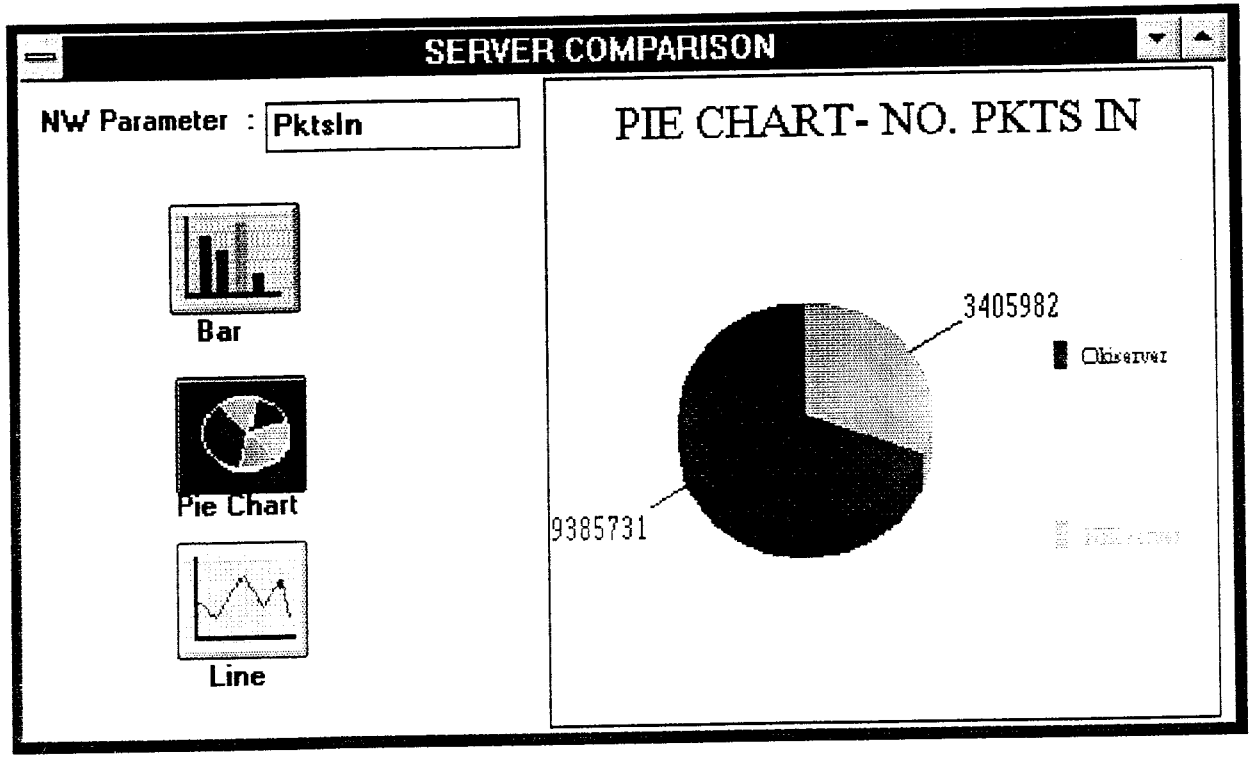

REAL MODE – CLIENT DETAILS
CLIENT NAME
ACTIVATED TIME
STATUS
node1 06:15:32 active
node2 06:16:12 active
rg3 06:16:55 idle
rem2 06:18:32 active
node4 06:19:01 active
rem4 06:19:21 active
SORT
CLIENTS
Sort Clients

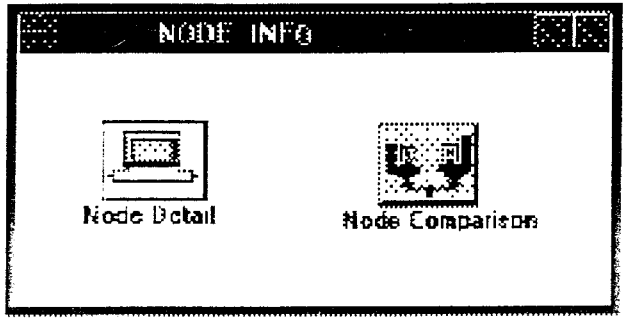

REPLAY MODE - CLIENT INFO
Options
File Selection
Network Summary
Node Summary
Sail NW Parameter
Select Parameter For Sorting :
BroadCast
Select Order Of Sorting
Ascending Order
Select File Name :
24031128.nw
24031111.nw
Select Server
OKServer
NECserver
Client List:
node1
root
node2
oki
node3
node4
root
node4

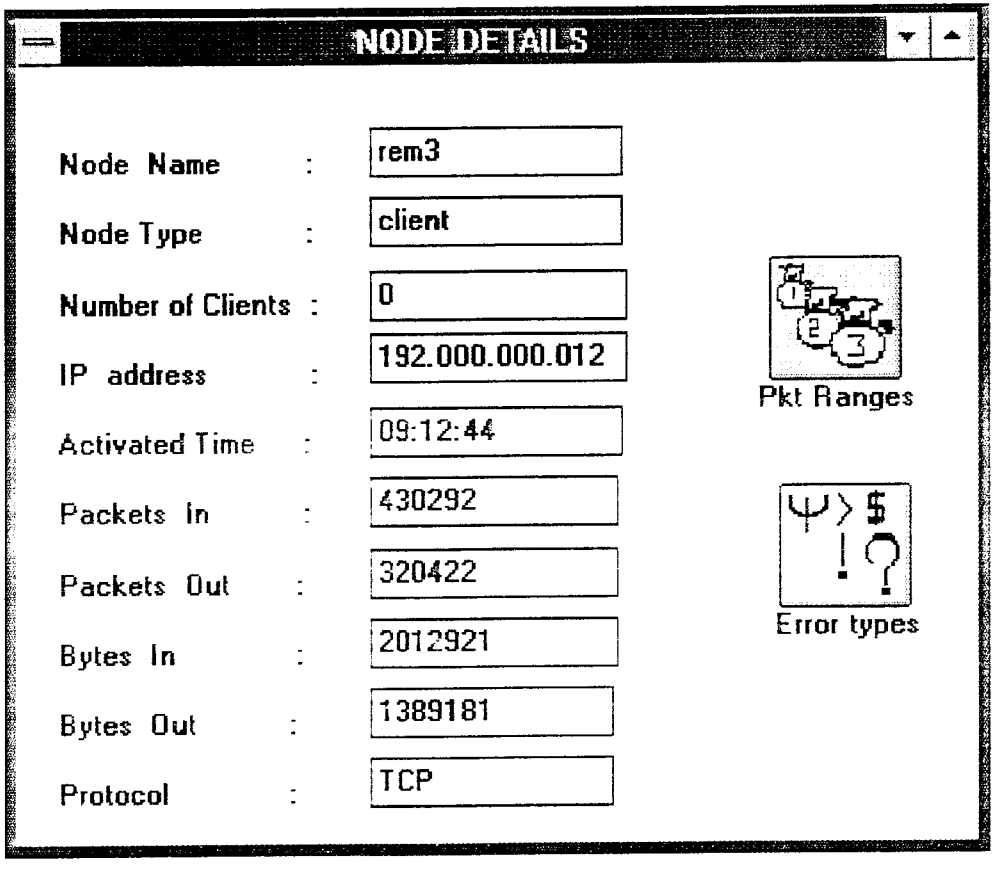

SORTING CLIENTS
NW Parameter
Traffic
Client Name
value
node3
14123
node1
10765
node2
10004
node4
20165
node5
32145
node3
49165
node9
52256

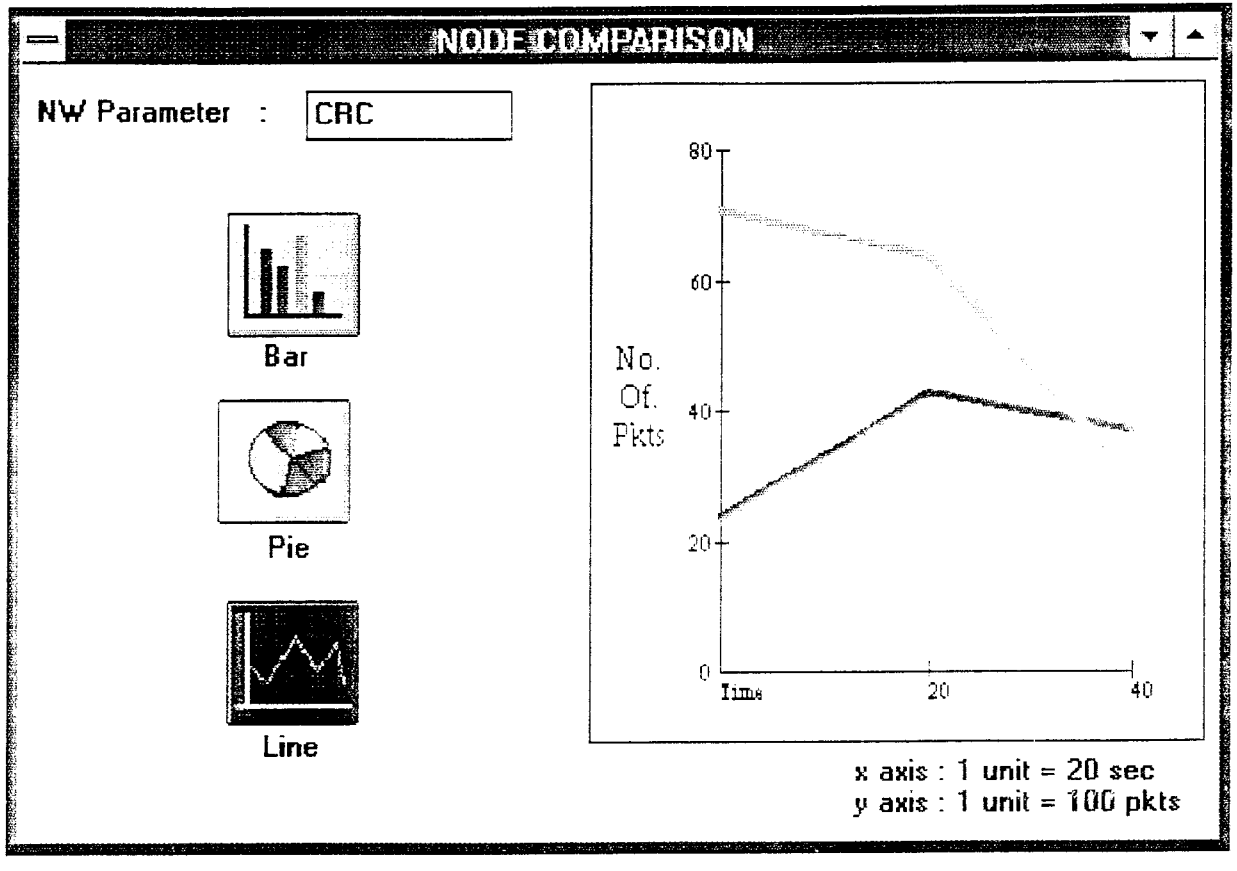

SERVER DETAILS
Server Name : Okiserver
Number of Clients : 8
IP address : 192.000.000.044
Activated Time : 08:40:21
Packets In : 654039
Packets Out : 432093
Bytes In : 4032949
Bytes Out : 3023984
Protocol : TCP
Packet Range
Error Types

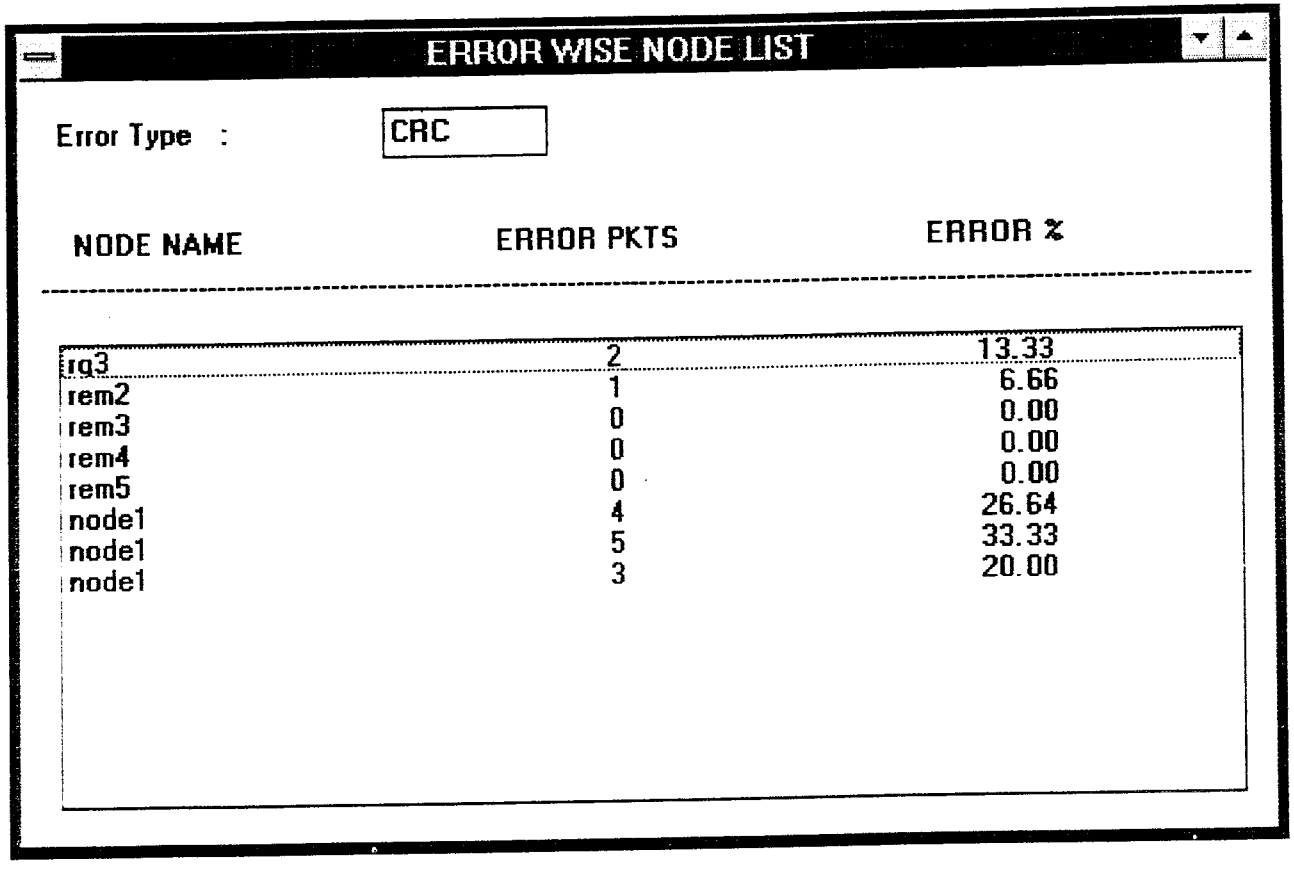

Server Comparison
Select A NW Parameter :
Number of BroadCast Pkts
Select Order Of Sorting :
Ascending Order

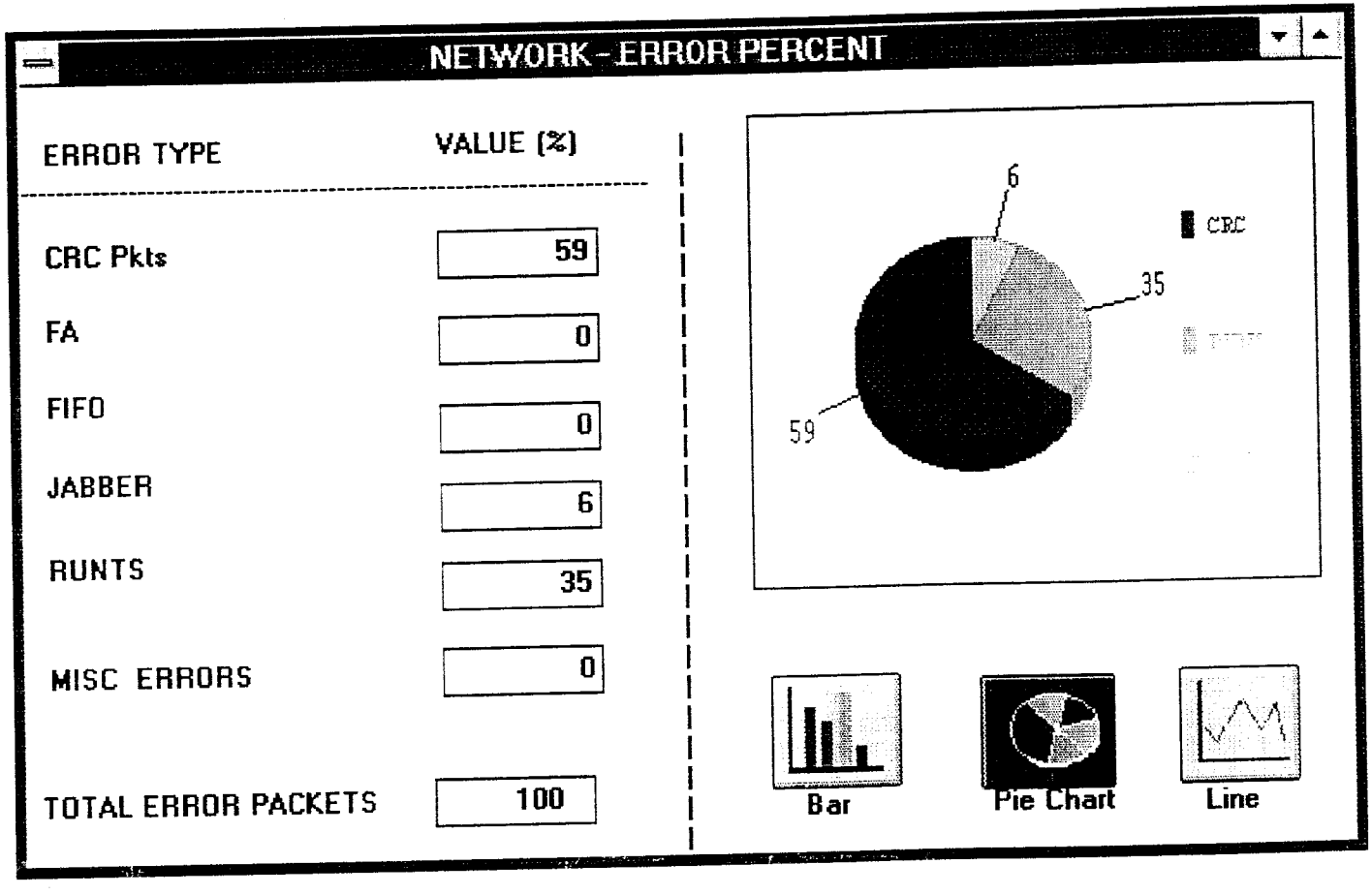

SERVER COMPARISON
NW Parameter : PktsIn
Bar
Pie Chart
Line
PIE CHART- NO. PKTS IN
3405982
9385731

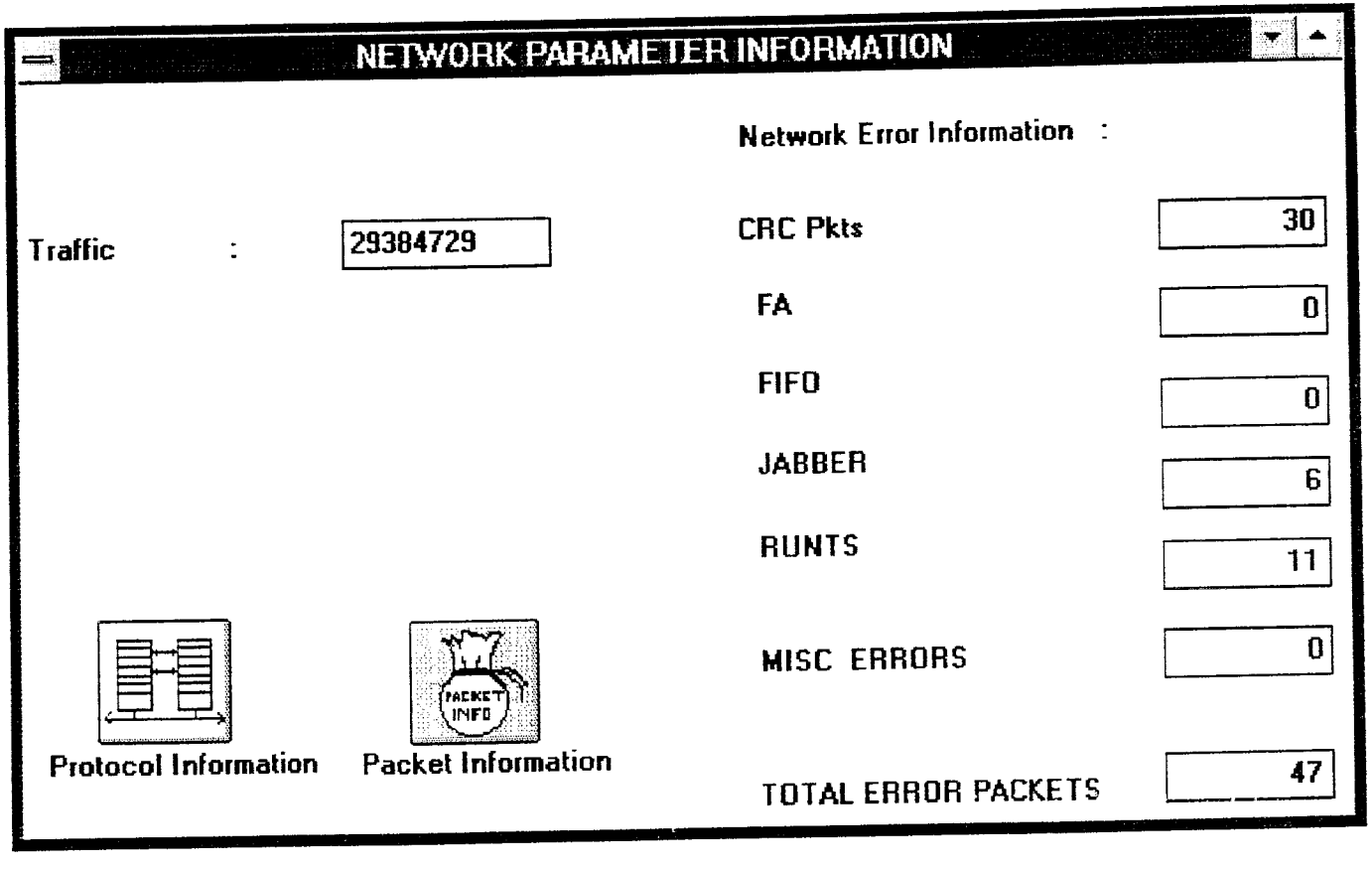

NODE INFO
Node Detail
Node Comparison

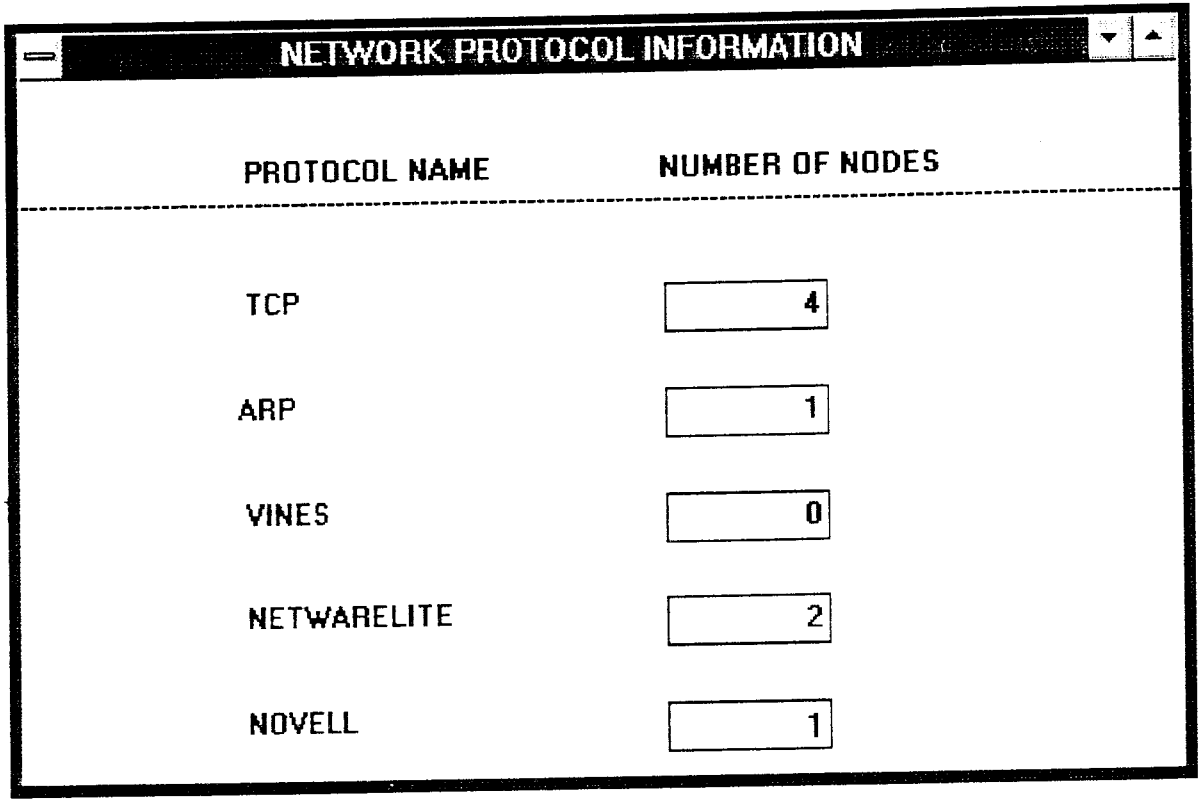

NODE DETAILS
Node Name : rem3
Node Type : client
Number of Clients : 0
IP address : 192.000.000.012
Activated Time : 09:12:44
Packets In : 430292
Packets Out : 320422
Bytes In : 2012921
Bytes Out : 1389181
Protocol : TCP
Pkt Ranges
Error types


NODE COMPARISON
NW Parameter : CRC
Bar
Pie
Line
80
60
40
20
0
No.
Of.
Pkts
Time
20
40
x axis : 1 unit = 20 sec
y axis : 1 unit = 100 pkts


ERROR WISE NODE LIST
Error Type :
CRC
NODE NAME
ERROR PKTS
ERROR %
rq3 2 13.33
rem2 1 6.66
rem3 0 0.00
rem4 0 0.00
rem5 0 0.00
node1 4 26.64
node1 5 33.33
node1 3 20.00


NETWORK - ERROR PERCENT
ERROR TYPE
VALUE (%)
CRC Pkts
59
FA
0
FIFO
0
JABBER
6
RUNTS
35
MISC ERRORS
0
TOTAL ERROR PACKETS
100
6
35
59
CRC
Bar
Pie Chart
Line


NETWORK PARAMETER INFORMATION
Network Error Information :
Traffic :
29384729
CRC Pkts
30
FA
0
FIFO
0
JABBER
6
RUNTS
11
MISC ERRORS
0
TOTAL ERROR PACKETS
47
PACKET INFO
Protocol Information
Packet Information


NETWORK PROTOCOL INFORMATION
PROTOCOL NAME
NUMBER OF NODES
TCP
4
ARP
1
VINES
0
NETWARELITE
2
NOVELL
1

9.5 ABBREVIATIONS

ARP	:	Address Resolution Protocol
CRC	:	Cyclic Redundancy Check
DBMS	:	DataBase Management System
DLL	:	Dynamic Link Library
FA	:	Frame Alignment
FCS	:	Frame Check Sequence
FIFO	:	First In First Out
GUI	:	Graphical User Interface
IP	:	Internet Protocol
LAN	:	Local Area Network
MAN	:	Metropolitan Area Network
MDI	:	Multiple Document Interface
MIB	:	Management Information Base
NIB	:	Network Information Base
OLE	:	Object Linking and Embedding
SNMP	:	Simple Network Management Protocol
TCP	:	Transmission Control Protocol
WAN	:	Wide Area Network