ACCESSING APPLICATIONS FROM MULTIPLE FILE SERVERS IN A NOVELL COMPUTER NETWORK

"Dissertation submitted in fulfilment of the requirement for the degree of Master of Engineering, University of Zambia."

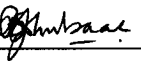
John Sherrin Isaac

UNIVERSITY OF ZAMBIA

1993

Declaration

I, Sherrin John Isaac, do solemnly declare that this dissertation represents my own work and that it has not previously been submitted for a degree at this or another university.

Signed: 

Date: 01 SEPTEMBER 1993

Approval

This dissertation by John Sherrin Isaac is approved as fulfilling the requirement for the award of the degree of Master of Engineering (M.Eng) by the University of Zambia.

Examiners' signatures:

1. Professor Fred Halsall, University of Wales, Swansea, U
2. Martin Hoby
3. Johann Marais
4. E-H. Jen

Date of Approval: _____

Abstract

This research focused on solving the problems of application access from multiple file servers in a Novell network that caters for a large user community. The application access technique discussed in this dissertation is a step towards creating a computer network system, called a Virtual Machine (VM), with the following characteristics: it is simple to implement, scalable, easy to manage, transparent to the user and suitable for the conditions prevailing in a university-type environment. The VM has some of the features of a distributed system, but is not as complex and is built around a very popular commercial network operating system called NetWare.

This dissertation presents the motivation, requirements, design considerations and performance of the developed VM.

TO MY PARENTS

Trademark Notices

3Com is a trademark of 3Com Corporation.

Apple is a registered trademark of Apple Computer, Inc.

C-scape is a trademark of Oakland Group, Inc.

EtherLink II is a trademark of 3Com Corporation.

IBM is a registered trademark of International Business Machines Corporation.

IBM PC, IBM AT, IBM XT are trademarks of International Business Machines Corporation.

INTEL is a trademark of Intel Corporation.

Internetwork Packet Exchange (IPX) is a trademark of Novell, Inc.

LAN Manager 2.0 is a trademark of Microsoft Corporation.

Macintosh is a trademark of Apple Computer.

Micro Channel is a trademark of International Business Machines Corporation.

MS-DOS is a trademark of Microsoft Corporation.

NetWare C Interface for DOS is a trademark of Novell, Inc.

NetWare is a trademark of Novell, Inc.

NFS is a trademark of Sun Microsystems, Inc.

NLM is a trademark of Novell, Inc.

Novell is a trademark of Novell, Inc.

Olivetti is a trademark of Ing. Olivetti & C., S.p.A.

Personal System/2 or PS/2 is a trademark of International Business Machines Corporation.

STREAMS is a trademark of AT & T.

Transport Layer Interface is a trademark of AT & T.

TTS is a trademark of Novell, Inc.

Tulip is a registered trademark of Tulip Computers International B.V.

UNIX is a trademark of AT & T.

UNIX system V is a trademark of AT & T.

UNIXWARE is a trademark of Univel.

VAP is a trademark of Novell, Inc.

Victor is a registered trademark of Victor Technologies AB.

X Window System is a trademark of the Massachusetts Institute of Technology.

Acknowledgements

I wish to thank the Department of Electrical & Electronic Engineering for providing the equipment and materials for this research and generously giving me leave to complete my research

Special thanks to my supervisor, Dr. J.S.J. Daka, for assistance, encouragement and valuable instruction during the course of the research.

I wish to thank Evans Chabala for helping me debug the main text and all my friends in Marshlands for urging me on.

Contents

List of Figures

List of Tables

CHAPTER ONE

INTRODUCTION

There are plans[1] to develop a campus wide computer network for the University of Zambia (UNZA) during the coming years. This network is expected to consist of a number of Personal Computer (PC) sub-networks that will connect to the main computer at the computer centre and to each other via a backbone network as shown in Figure 1.1. Each sub-network will consist of PCs and file servers and will connect to the backbone by a bridge[2].

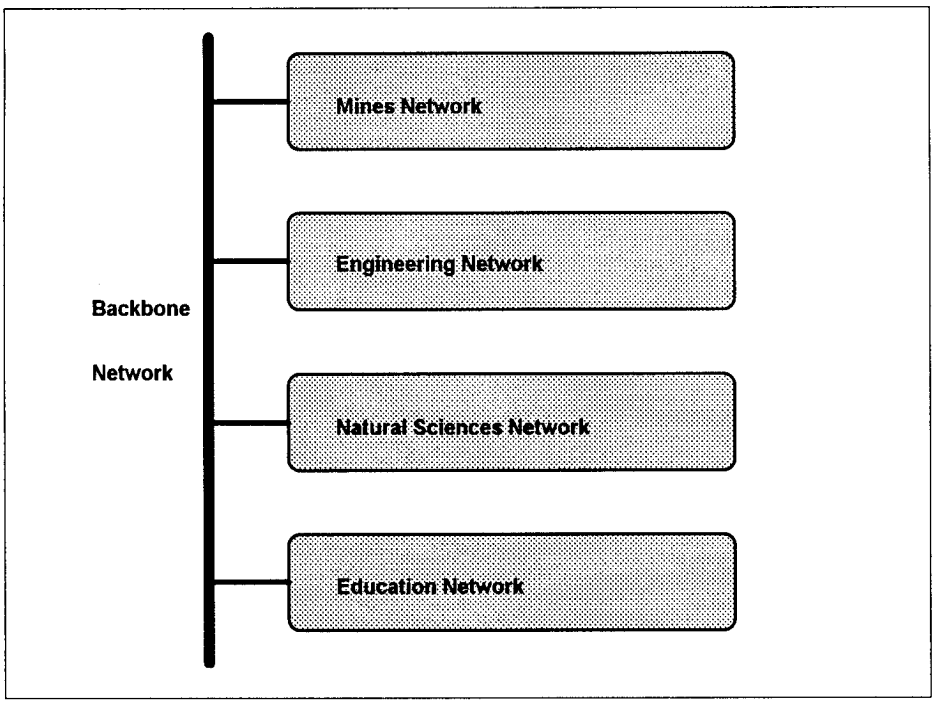


Figure 1.1: Expected UNZA Sub-network Layout

This research started in view of the anticipated development. The main objective of the research is to write software that will enable a transparent (i.e. invisible) computer network for MS-DOS based PCs, suitable for the conditions prevalent in an environment like a university to be built. The computer network used in the research is known as a Novell network and the focus of the research is to enable users to transparently access applications located on various file servers in this network. There are problems associated with transparent application access, but before stating these problems the following terms are introduced: computer networks, distributed systems, virtual machines and NetWare.

1.1 Computer Networks

A computer network is an interconnected collection of autonomous computers. Some of its uses are:

1. *Sharing Resources.* It is quite expensive to provide printers for each stand-alone PC. Many networked PCs can share one printer. Similarly, everyone can share programs, data and other equipment on the network.
2. *Reliability.* To circumvent hardware failures all files could be replicated on a few machines so that if any one of them is unavailable, the other copies could be used.
3. *Performance.* Computer hardware is becoming cheaper every year. As a result of this, a collection of small computers now performs better for their price than a single large computer.
4. *Expansion.* An increase in the demand for computing resources can easily be met by adding more computers to the network. If a large multi-user computer becomes overloaded, it usually has to be replaced at great expense and considerable disruption of service (it is not easily expandable).
5. *Communication.* You can send electronic mail messages to one another quickly over a network. It is also possible for people to work together on projects though

physically separated by distance. Databases can also be interrogated remotely over the network.

A computer network that covers a small geographical area is known as a Local Area Network (LAN). A LAN, for example, connects all the computers situated in one campus of a university. LANs can be interconnected to form a Wide Area Network (WAN).

Current LANs are in two varieties: the workstation/server type, and the serverstation type. The first type classifies the computers on the network as being in two mutually exclusive categories: servers and workstations. Workstations have very little interaction with each other and do not provide any service to other workstations. All workstations access the same set of servers. An example of this in PC networks is Novell's NetWare 3.11 operating system. In the second type, every computer on the network can act as a server for the others. Accordingly, each computer has both server and workstation software. The serverstation type is commonly known as a peer-to-peer[3,4] LAN. However, this term could be mis-leading, because processes in two computers in both types communicate on a peer-to-peer level as far as the International Standards Organisation (ISO) Open Systems Interconnection (OSI) reference model[5] is concerned. An example of the serverstation type in PC networks is Microsoft's LAN Manager 2.0. Most UNIX based networks are also of this type.

In the serverstation type, management of user accounts and data will become increasingly difficult as the network grows in size. Eventually, some machines that will only act as servers will be designated. At this stage it will imitate the workstation/server type of LAN.

Although computer networks are very useful, and a lot of developments have occured in this field, the main problem with current computer networks is that they are not transparent; the users are aware of other computers on the network. The user logs into one machine explicity and uses that machine only, until doing a remote login to another machine. Basically, the user handles much of the network management and this is

difficult for inexperienced network users. Making the network transparent will remove this complication for the user. Network transparency is the speciality of distributed systems.

1.2 Distributed Systems

The interpretation of the term 'distributed systems' differs in literature[6,7,8,9]. I will use the following explanation:

A distributed system is a special type of network whose software gives it a high degree of cohesiveness and transparency. The existence of multiple autonomous computers is transparent to the user. The operating system finds the best processor for a job and does all the management of file traffic to and from that processor. The system also performs all other system functions automatically and does not require the user's knowledge. In other words, the user does not explicitly have to login to computers where the applications or files he is interested in reside, and does no network management.[7]

A distributed system thus also includes distributed processing. With this type of system it is even possible to achieve a form of parallel processing. Here the operating system creates the impression of a single computing environment even though there are multiple processors on the network. The software for a distributed system as a result has to be very complex. In making things easier for the user, the complexity has been shifted to the network management software. Very few distributed systems have been designed and implemented. Even fewer are actually used.

Although a user will find a distributed system to be more useful than a plain computer network because of the increased transparency and especially the capability of transparent distributed processing, it is this same capability that makes it very complicated and difficult to implement. A simpler alternative to distributed systems, called virtual machines, is presented in the next section.

1.3 Virtual Machines

A degree of network transparency can be achieved by presenting the user with the illusion of being on a very powerful stand-alone machine that contains all the applications and data files he is entitled to, and is connected to all the peripherals he is allowed access to. This powerful stand-alone machine does not, of course, exist and can therefore be termed a Virtual Machine (VM). Various file servers in the network store all the files he requires and has, attached to them, the various peripherals he is interested in.

The purpose of introducing this new term is to classify systems that lie between a plain computer network and a distributed system with respect to complexity. A VM does not have distributed processing capabilities. In a VM, during terminal emulation, an application might run on another machine but the load is not split between many processors. This difference distinguishes a VM from a distributed system. Software that presents a VM to the user can therefore be simpler than the operating system software for a distributed system. VM software can be built around existing operating systems, even network operating systems. In this way, a useful network system that system designers and managers find uncomplicated and simple to implement can be devised. The choice of a good network operating system will obviously ease the design and increase the acceptance of a VM. The network operating system used in the research is introduced in the following section.

1.4 NetWare

NetWare[4] is a network operating system produced by Novell. 'Novell Network' is a popular term for a NetWare based computer network. There are currently a few versions of NetWare that are available commercially. NetWare versions 3.10 and 2.15 were selected as the base network operating systems for this research. The reasons for choosing NetWare and these particular versions are:

1. Novell is the market leader in MS-DOS based networking.

2. Novell is committed to the OSI philosophy.
3. These are the network operating systems available in the School of Engineering.
4. These versions of NetWare are based on the workstation/server type of relationship.
5. NetWare provides a fairly transparent interface to users.

The above mentioned reasons are explained below.

The primary PC operating system at UNZA is MS-DOS. There is an existing investment and there is an expected future investment at UNZA for this operating system. Even though this is the case, the ability to extend the network to include systems with other operating systems in future is important. The commitment to the ISO OSI philosophy means that the Novell network can be connected to other networks to form an inter-network. Since Novell is the market leader in MS-DOS based networking, many gateways are available that connect Novell networks to other types of networks. Many other products are also commercially available for this environment.

The development and testing of software for the research was done at the School of Engineering, UNZA. This school has a Novell network with two file servers. One file server has NetWare 3.10 as its operating system and the other runs NetWare 2.15. Research costs were reduced considerably because these systems were available for the research. (Actually, any version of NetWare more recent than 2.10 could have been used for the research).

The workstation/server relationship of NetWare enhances security and provides for an easy to maintain system. All files are assumed to reside on file servers. Compared to the total number of PCs on the network, the number of file servers is quite small. All installation of applications is done at the file servers. Legitimate users of the network are defined only at the file servers and all authentication is done there as well.

NetWare hides the existence of multiple sub-networks from the average user. Logging in to a server on another sub-network is no different from logging in to a server locally. The file server's hard disk can be made to appear as another local PC disk drive. Almost all MS-DOS operations can be performed on the files in this disk drive. Remote printers (usually attached to file servers) can be made to appear as local printers. This feature of NetWare hides some of the complexity of a computer network. Logging in to a file server is, of course, not transparent to the user.

1.5 Problems Associated With Application Access in a Multiple File Server Novell Network

Computer networks are very useful but unfortunately the current generation of computer networks are not transparent[7]. Distributed systems address the issue of transparency in computer networks, but are very complicated and difficult to implement because of the distributed processing aspect of it. Furthermore, most research in the area of distributed systems has been conducted in the UNIX environment[6,10,11,12].

With the availability of cheap and powerful PCs, it now looks attractive to base a campus network primarily on these machines. There has not been much research done for PC based campus computer networks. It is only in recent years that PC sub-networks have started appearing in many universities. Generally, these PC sub-networks are based on a commercial PC network operating system such as Novell's NetWare. Usually, only one file server will be found in this PC sub-network and the users are provided access to a few applications on it. If the number of users, applications and file servers in the system increase, then users and network managers will encounter some problems described later in this section. These problems partly arise because of the market that PC networks traditionally cater for — the business sector. In the business sector it is not common to find a large user community, the membership of which is constantly changing, nor is it common to find all users needing access to all file servers. Normally, the accounting department in a company (for

example) would have a few of their personnel defined on a file server containing a few applications relevant to that department. A similar situation would be found in the engineering department, and so on. Very few users would need access to more than one file server.

In a university environment (could also be true of some business concerns), if the computer network is almost entirely PC based, there would have to be many file servers on it just because of the enormous volume of data generated, and the number of applications and users. All the users will need access to all the file servers if unnecessary duplication of applications and data is to be avoided. In this situation a product like Novell's NetWare, where users are created on the file servers, poses a few problems in enabling users access information or applications from multiple file servers (described further below). Even if NetWare develops to solve the problem caused by having users created on the file servers, the problem of application access remains. A transparent network file system will not solve this problem though it will make it easier to solve it. Accessing an application does not mean merely locating it, but includes starting it. To start an application, many things sometimes needs to be done. The MS-DOS 'path' variable needs to be set, and other environmental variables might also have to be set before an 'executable' file can be run.

This research tackles the following problems that occur when users require access to various applications stored in different file servers in a Novell network (NetWare versions 3.11 and earlier versions):

1. *The user's problem.* He must know which file server has the application he requires. Then he must log into that file server; if he does not have a login name for that server or he does not know the password, it will not be possible to access the application required. Furthermore, after logging in, the user will have to know how to access the application (which directory it is in and how to start it). If the location of an application is changed then the user will have the same problems all over again.

2. *The file server administrator's problem.* To illustrate this problem, suppose there are 3000 users and eight file servers. Also suppose that each user has to be able to login to every file server, and different categories of users should have different access rights to the applications and data stored in those file servers. Currently, the file server administrators can employ two schemes to achieve these goals.

   First scheme:

   The user population is divided into eight groups. Each group is then created on only one file server. Every user is then supplied with two usernames; each with their own associated password. The first username is unique to the user and he can change the corresponding password. The second username is common to all users throughout the network who are in the same category as this user. The user can not change the password belonging to the second username. With the first username, the user has access to all the applications he is entitled to on his 'home server'. He will also have access to his personal data files. With the second user name and password, the user can access applications that are resident on the other file servers.

   The problem with this scheme is that it is not very secure. All the users in a particular category have to be informed about the password related to the username of that category. This information could easily fall into the 'wrong hands'. If the server administrator changes the password, this will again have to be communicated to all the affected users.

   Second scheme:

   Each one of the 3000 users is created on each one of the eight file servers. This is the standard method employed in most Novell networks with the requirements given in this illustration and has better security than the former scheme.

   The problem with this scheme is that the labour requirements in creating and maintaining 3000 users on each file server is not minimal. Also, the users have to

remember the names of all the file servers and their usernames and passwords for these file servers. The usernames on all the file servers might not be the same for the same user because a typing mistake might have occured in the creation of the user's account on a file server. This will create a lot of confusion for the user. The probability of a user forgetting the password for one of the file servers is also very high. This will also lead to more work for the server administrators to set things right and to reset passwords and other things of that nature.

1.6 Research Aim

In the preceding sections the proposal that a VM could be both useful and simple to implement has been advanced. The VM can be built around Novell's NetWare for a PC based network. However, the problems mentioned in section 1.5 (which appear not to have been solved in the past) must first be solved. This research aims to solve these problems and will thus be a step towards the creation of an easy to manage VM. The peripheral attachment aspect of the VM is partly solved by NetWare, but awaits a more manageable solution.

The requirements of the VM being developed are:

- can run existing applications for stand-alone machines.
- is primarily designed for the PC environment.
- provides a transparent network to the user.
- scalability: allows easy maintenance of system data and management and should facilitate the addition of more PCs and users with a minimum of management and disruption to service.
- station mobility: allows PCs to be moved around on the network without reconfiguring VM software.
- user mobility: allows any user to use any computer on the network.

1.7 Analysis of Chapters

Chapter two presents information on distributed systems; models, examples and design issues. Chapter three provides an overview of the developed VM; how the aims of the research have been realised and how the design issues of distributed systems that are relevant to the VM have been addressed. Chapter four presents some details of the VM design. Chapter five provides some information on the performance of the VM. Tests conducted on the system are described and the results are presented and discussed. What some people say about the developed VM is also mentioned. Chapter six presents the conclusions of the research and suggests areas for future work.

CHAPTER TWO

DISTRIBUTED SYSTEMS

A relationship exists between distributed systems and VMs — both have transparency as a central goal. Distributed systems can be considered to be a superset of VMs because they also include distributed processing. Since the two systems are related, a study of distributed systems is useful in the design of the VM. This chapter looks at some models of distributed systems and two successful implementations (one is very similar to a VM and the other is a true distributed system), and then compares these two examples with the VM that has been designed. In this way, the scale and complexity of the examples as opposed to the developed VM (along with other differences) are emphasized. Some of the important issues in distributed systems are also relevant to VMs and are described as well.

2.1 Models of Distributed Systems

Three architectural models of distributed systems have been identified by Coulouris and Dollimore[8]. These are outlined in sections 2.1.1 to 2.1.3.

2.1.1 Workstation/Server Model[8]

The majority of distributed systems in use and under development are based on this model which is shown in figure 2.1. Each user is provided with a single-user computer known as a workstation. Application programs are executed in the users' workstation (note: this is not a true distributed system as explained in chapter one). The

workstations are integrated by the use of networking software which enables them to access the same set of servers. The servers provide access to shared resources.

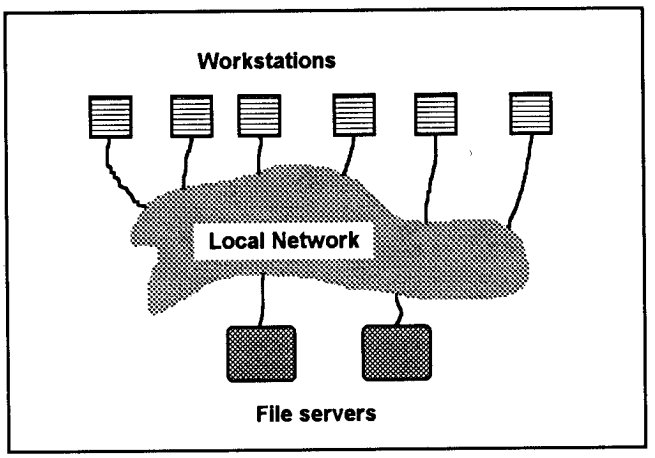


Figure 2.1: The Workstation/Server Model[8]

2.1.2 Processor Pool Model[8]

Programs are executed in a set of computers managed as a processor service. In the pure processor pool model, users are provided with terminals, rather than workstations. A hybrid model containing workstations has emerged due to the wide availability of workstations. The hybrid model is shown in figure 2.2.

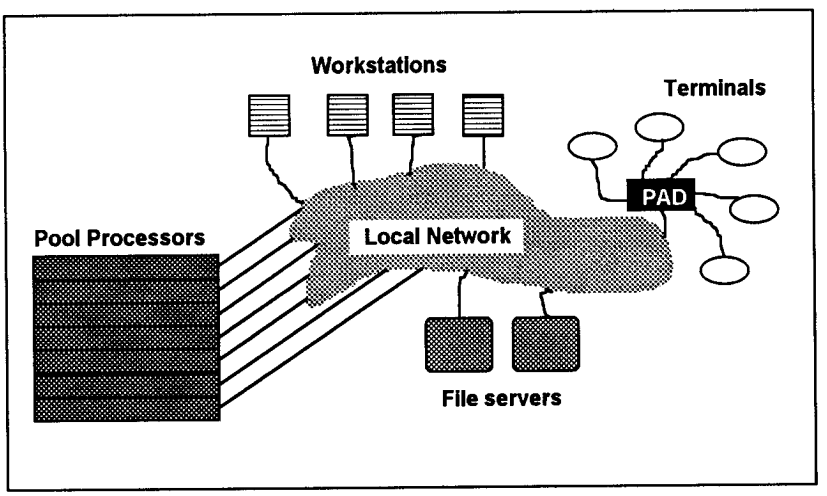


Figure 2.2: The Hybrid Processor Pool Model[8]

The hybrid model is similar to the workstation/server model. The pool computers can be viewed as a processor service, in which the pool computers are available for dynamic allocation of tasks that are too large for workstations or tasks that require several computers concurrently. This is a true distributed system.

2.1.3 Integrated Model[8]

This brings many of the advantages of distributed systems to heterogeneous networks containing single-user and multi-user computers. In this model each computer has software to perform both the role of server and application processor. The operating system software located in each computer is similar to that of a centralised multi-user system, with the addition of networking software. This model is shown in figure 2.3 and is suited to most campus networks because of the prevalence of mini and main-frame computers.

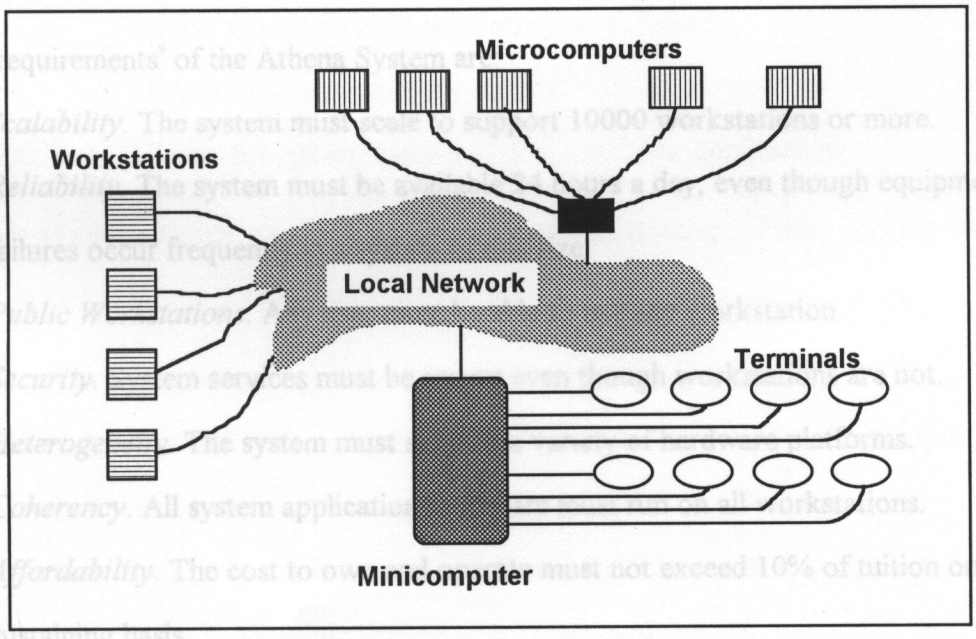


Figure 2.3: The Integrated Model[8]

2.2 Examples of Distributed Systems

Two successful implementations of distributed systems are described below. It should be noted that the first example described does not really fit into the category called distributed systems because it does not include transparent distributed processing. In this respect it is closer to a VM. The systems presented below are quite simply huge, and need complex management systems for its operation. A lot of research, time and money were invested in these projects (by teams of very competent people) before their eventual implementation.

2.2.1 Athena

Project Athena[6] was established in 1983 at MIT to develop high quality computing based on a large number of workstations. Athena is one of the largest centrally managed educational workstation networks. It is based on the workstation/server model of a distributed system.

The requirements[6] of the Athena System are:

- *Scalability*. The system must scale to support 10000 workstations or more.
- *Reliability*. The system must be available 24 hours a day, even though equipment failures occur frequently in a system of this size.
- *Public Workstations*. Any user must be able to use any workstation.
- *Security*. System services must be secure even though workstations are not.
- *Heterogeneity*. The system must support a variety of hardware platforms.
- *Coherency*. All system applications software must run on all workstations.
- *Affordability*. The cost to own and operate must not exceed 10% of tuition on a sustaining basis.

By September 1990, the system had:

36	file servers
79	postscript printers
3	name servers

3	post office servers
2	authentication servers

In terms of disk storage:

40	Gbytes on workstations
50	Gbytes on network file servers
50 - 100	Mbytes on each workstation.

Some other system statistics:

1000	workstations (will scale to 10000)
40	clusters
10 - 120	workstations/cluster
10000	active user accounts
4000	logins/day
5	operations programmers for 1000 workstations
3	system managers/1000 workstations (in future)
20	Mbytes of source code for Athena
400	Mbytes for Athena system — Unix, Athena, applications

Athena system services are: name service, file service, printing service, mail service, real time notification, service management, authentication, installation and update.

Other information concerning Athena's operation:

1. Athena performs all processing on the local workstation as a default. Remote execution can be initiated manually through the X Window System.
2. Athena concentrated on distributed services and did little with end user services.
3. Athena provides its distributed services on dedicated servers.
4. Athena uses UNIX access control and supports a system wide name space.
5. Athena uses conventional subroutines and does not support process migration.
6. Athena uses a single operating system programming and user interface (BSD 4.3).
7. Athena has about 10000 file roots for flexibility and reliability.

2.2.2 Amoeba

The Amoeba 4.0 system[10] was released in 1990. The Amoeba project had been under way for nearly 10 years by that time. This is what the creators of the system say about it:

> We envision a system for the 1990s that will appear to the user as a single, 1970s centralized time-sharing system. Users will not know which processors their jobs are using (or even how many), where their files are stored (or how many replicated copies are maintained to provide high availability), or how processes and machines are communicating. All resources will be managed completely and automatically by a distributed operating system.[10]

Amoeba is a hybrid pool processor distributed system. This model allows a system to be built in which the number of processors greatly exceeds the number of users. Typically, the processor pool consists of many single board computers, each with several megabytes of private memory and a network interface. Processors can be added as the user population grows. A degree of fault tolerance to processor crash has been provided.

"Amoeba's system interface is quite different from those of today's operating systems."[10] This is because most operating systems are not designed for distributed systems. "Building directly on the hardware instead of on an existing operating system has been absolutely essential to Amoeba's success. A primary goal was to build a high-performance system, and this can hardly be done on top of another system."[10] There is an emulation package that allows most UNIX utilities to work on Amoeba, sometimes with small changes. This bridges the gap with existing UNIX systems.

Users are intended to view Amoeba as a collection of processes and information objects maintained by servers. The objects and servers are identified by sparse capabilities. A capability allows its possessor to perform certain operations on the object it names. Security of capabilities is based on their being hard to guess. Several

different file systems have been built in Amoeba, including a UNIX file system with UNIX system calls, a flat file service and an advanced transaction-based file service called the Free University Storage System[6].

Some local Amoeba systems in several countries in Europe have been interconnected to form one extended distributed system[6]. Amoeba research focuses on the use and management of the processor set[6].

2.2.3 System Comparisons

The two systems described above have some similarities with the VM that has been designed. All have almost similar objectives.

The VM and Athena are both campus wide systems based on the workstation/server model (though Athena has some aspects of the integrated model). Like Athena, the VM is built around an existing operating system. All processing is performed on the local workstation as a default. Services are provided on dedicated servers. The VM has a database server which serves information to requesting clients. Decisions are made by the client software. This arrangement performs a similar function to Athena's name server. Unlike Athena, authentication is handled by the file servers. The VM uses NetWare access control[13] not UNIX. Special service management software is not necessary as the VM is less complex than Athena. The VM is entirely PC based whereas Athena has a variety of hardware platforms.

Amoeba has been designed for environments larger than a single campus[6]. It is not based on the workstation/server model of distributed systems and is not built around an existing operating system. Amoeba dynamically assigns network processors to tasks as the processing load increases. The VM does not have any distributed processing capabilities. All processing is done on the local PC.

2.3 Issues in Distributed Systems

2.3.1 Transparency

Transparency is defined as the concealment of separation from the user and the application programmer, so that the system is perceived as a whole rather than as a collection of independent components.

Eight forms of transparency have been identified[8,14].

1. *Access transparency* enables local and remote files, devices and processes to be accessed using identical operations.
2. *Location transparency* enables objects to be accessed without knowledge of their location. This means that objects can be moved without changing its name.
3. *Concurrency transparency* enables several users or application programs to operate concurrently on shared data without interference between them.
4. *Replication transparency* enables multiple instances of files and other data to be used to increase reliability and performance without knowledge of the replicas by users or application programs.
5. *Failure transparency* enables the concealment of faults, allowing users and application programs to complete their tasks despite the failure of hardware or software components.
6. *Migration transparency* allows the movement of objects within a system without affecting the operation of users or application programs.
7. *Performance transparency* allows the system to be reconfigured to improve performance as loads vary. The overhead associated with referencing remote resources should be so small that it can be ignored.
8. *Scaling transparency* allows the system and applications to expand in scale without change to the system structure or the application algorithms.

2.3.2 File Service

This deals with the presentation of files to a user so that the user can access these files transparently. In a stand-alone workstation, the operating system would present the user with the directory structure of the local hard-disk. If the user can not find the file he is interested in within the current directory, then he will continue his search through the directory structure of the local hard-disk (and if unsuccessful, then through the directory structure of a floppy disk). All operations on the files in this local system will be familiar and consistent to the user. A good file service in a distributed system will enable a user to access files resident on another computer using the same operations that he is used to on the local workstation.

2.3.3 Naming

This important issue deals with how to name an object and how to locate an object from its name. The location is usually determined from the name by an algorithm, a table lookup, or a combination of the two. The benefit of a name service is that even if the address of an object is changed, as long as the name remains the same it can be accessed without the accessing entity being aware of the change in address. In other words, a name service enables objects to become location transparent.

In the table lookup case, distributed database issues have to be considered. Should the database be centralised? If not, should the database be partitioned or replicated? How is the database to be updated?

2.3.4 Scalability

A scalable distributed system easily copes with the addition of stations and users to the network with a minimum of management and disruption to network services[11]. It is not wise to store information pertaining to the physical aspects of the network and software (such as the locations of the servers and applications) on all the stations on the network because storage and labour requirements would increase and thus affect scalability. Also, broadcast protocols should be used with care in a distributed system

because they do not scale well[6]. Generally, a good design of a distributed system should ensure that scarce resources (such as storage or manpower) do not increase linearly with the number of stations or users.

2.3.5 Security

Authentication and authorisation are important concerns in the design of a distributed system. Authentication deals with establishing the identity of the object requesting the service. Is it what it claims to be? A measure of authentication can be achieved using passwords and a password-checking service.

Authorisation has to do with user access rights and security clearance to perform certain operations. Authorisation can be performed by the servers using access control lists showing membership in groups[6]. Another way is to use capabilities as in Amoeba[10].

2.3.6 Compatibility

This issue arises because of the existence of differing hardware and operating systems on the network. For example, an application might run on one computer and not on another. At least three levels[6] of compatibility are possible. These are the binary, execution and protocol levels of compatibility. In binary compatibility all processors are able to execute the same binary instructions. Execution compatibility exists between two systems if the same source code can be compiled and executed properly on both. Protocol compatibility requires all system components to support a common set of protocols and is the most flexible form of compatibility.

CHAPTER THREE
VM MODEL

This chapter gives a survey of the VM; the general methods by which the objectives of the research have been realised and how the various issues in distributed systems mentioned in chapter two have been addressed.

3.1 Components of the VM

3.1.1 Hardware Components

The VM has the following major components:

1. *Network hardware*: network cards for the PCs, network cables, terminators, etc.
2. *File servers*: these are PCs with large hard disk capacities and at least 4 MB of RAM. File servers are usually machines that have been dedicated to serve files to requesting workstations.
3. *PCs*: these are at least IBM XT class machines, but more usually IBM AT class machines. PCs usually have a small local hard disk and at least 640 KB of RAM.
4. *VM database servers*: almost any PC with at least 1 MB of RAM can act as a VM database server.

The suggested layout of the UNZA VM (consisting of a sub-network with at least one file server in each school) is given in Figure 3.1. It would be wise to connect the sub-networks to the backbone using only bridges since most network traffic is expected to be 'local' to the sub-network. Bridges will also localise problems to a sub-network

caused, for example, by a misbehaving node transmitting spurious broadcast packets. These spurious broadcasts can bring down the sub-network because receiving PCs would soon run out of buffers and if the sub-network is on an Ethernet, none of the PCs would be able to connect to file servers or even maintain established connections. Though the sub-network will come down, at least the entire network will not come down.

(The VM would still be operational even if there are no sub-networks or bridges or if the network is arranged in a different configuration from the one shown in Figure 3.1.)

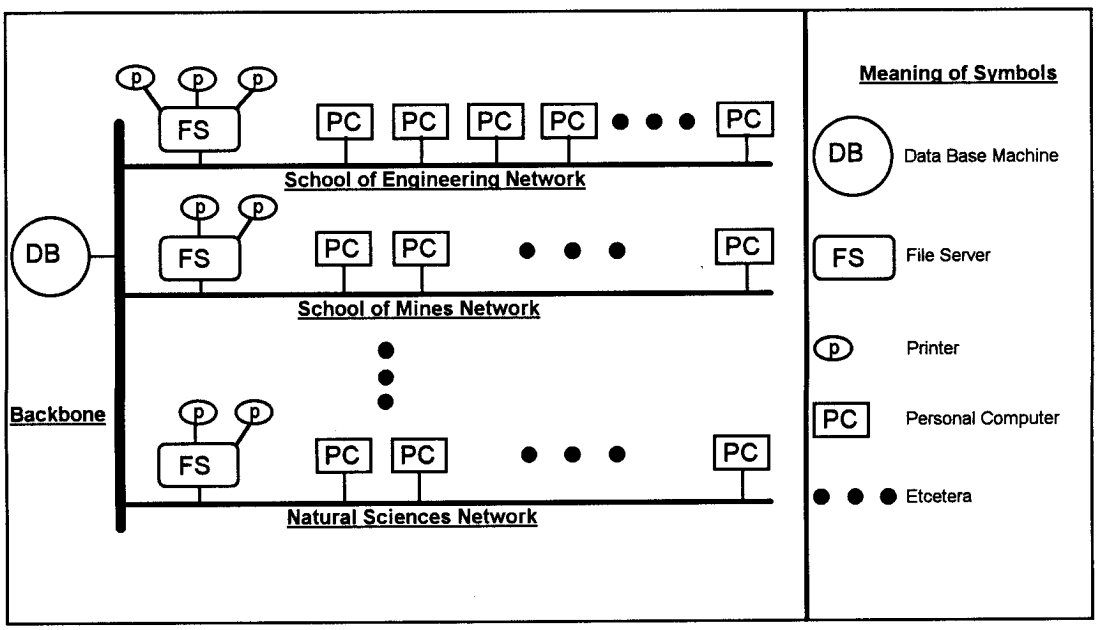


Figure 3.1: Suggested UNZA VM Configuration

3.1.2 Software Components

The VM system software has the following main components:

1. *Novell NetWare*: This really consists of two[4,15] main pieces of software. One is an operating system for a PC that makes that PC act as a file server for the rest of the network. The other part is called the NetWare shell and runs on all the other PCs on the network. This shell enables the PCs to communicate with each other and the

file servers on the network. The shell also makes the file server's hard disk appear as a local disk drive and makes resources such as printers that are attached to the file server appear to be local devices. The NetWare shell consists of two Terminate and Stay Resident (TSR) programs called 'IPX.COM' and 'NETx.COM'. 'IPX.COM' handles the PC's communication on the network and is hardware dependent on the network card that has been installed on the PC. 'NETx.COM' is dependent on the operating system on the PC and handles network requests. The 'x' in 'NETx.COM' is replaced by the version number of the MS-DOS operating system on the PC. If MS-DOS[16] version 5.0 is used, then 'NET5.COM' has to be loaded into memory on the PC.

2. *Database for the operation of the VM.* An important requirement of the VM system is that system data should be easy to maintain and manage. Therefore it is unwise to store information in the individual PCs about the physical aspects of the network and the software. Instead it is desirable to have all data concentrated in a single database that will be used by all the PCs. In this way all management is done within the database and not in the PCs.

   The database for the operation of the VM is of two parts; the first is located on all the file servers in the network (this is really the NetWare database) and the other (called the VM database) is located on PCs designated as VM database servers. This arrangement could bring the entire network down if the VM database server fails. Some network resiliency to such failure can be made by replicating the VM database on various servers in the network. When the VM database has to be updated, each VM database server is brought down (one after the other) and the new VM database is loaded and the server restarted. The database contains information on:

   - Users (access rights, password, etc.). This is a distributed database located in the bindery[17] of all the file servers and is non-replicated.

- Software (file servers on which it is stored, and how to start it, etc.). This is a database located on the VM database server machine and is replicated on all the database server machines.
- Application files. This is a distributed database located in the file server's directory structure and is replicated on a few file servers.
- Menus. The content of menus and the relationship between the menus is a database located on the VM database server machine and is replicated on all the VM database server machines.

3. *VM database server program*. This is a program called 'DB.EXE' that runs on the VM database server machine and makes the information in the VM database available to requesting PCs. The request for data and the reply to the request follow strict formats. All requests for data from the PCs are queued and serviced on a first-come-first-served basis. When the VM database has to be updated, the database administrator can run a specially developed program called 'LOADDB.EXE' that will automatically bring down all the VM database servers (one after the other) and load the new VM database from a specified location, and then bring up the servers again.

4. *VM PC Software that is to be loaded on each PC*: The main PC software that has to be loaded into each PC to enable the operations of the VM is called 'VM_MENU.EXE'. This is loaded into the PC each time the user wants a menu of applications and types 'VM'. 'VM' is a batch file that is available to all users after logging in to the system, and this batch file invokes 'VM_MENU.EXE'. 'VM_MENU.EXE' communicates with the VM database server to present the menus to the user. 'VM_MENU.EXE' also checks the bindery of the user's home file server to determine the user's access rights to the applications and logs users in to appropriate file servers where the desired applications are stored. There are other utility programs that have been developed for the smooth operation of the VM and that run on the local PC, but the VM PC programs that the user will be

aware of are 'LG.EXE' to login to the network, 'VM' to access the menu, and 'LGO.EXE' to logout of the network.

From the user's viewpoint, the VM in Figure 3.1 will appear as shown in Figure 3.2. The existence of many sub-networks and file servers is hidden by a combination of NetWare and the VM software. The VM software interacts with the file-servers and the VM database server to create the illusion shown diagrammatically in Figure 3.2. An inexperienced computer user might not even be aware of the network or the file server.

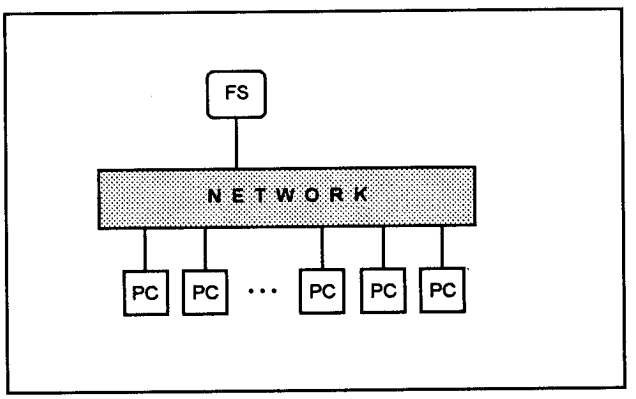


Figure 3.2: Presentation of the Network to the User

3.2 The Application Access Technique

The basic VM model involves a user logging into their home file server from any PC on the network. All authentication is handled by the home file server. Whenever the user wants an application, a menu program found on the search path of the home file server is loaded into the PC's memory and invoked. This program contacts a VM database server which is advertising its services on the network and various menus from the VM database server are presented to the user. By stepping through the menus, the user selects the application that he requires. The VM menu program contacts the relevant file server that has this application (also from information

obtained from the VM database server) and subsequently the VM menu program terminates and the application is started.

All communication between the VM menu program and the VM database server are by defined protocols and use NetWare Application Programming Interface (API) calls[13] for MS-DOS. The applications are started by batch files that set all necessary environmental variables and search mappings before starting the application. When the application terminates, all the set environmental variables are reset and search mappings are deleted.

The user is given access to all file servers by means of a common login name and password that is provided for all users of a particular VM category. The login name and password are obtained from the VM database server by the VM menu program, and the actual login to file servers other than their home file server (and logout) is performed on behalf of the user by the VM menu program.

3.3 Transparency

- Access transparency: This is discussed under 'File Service'.
- Location Transparency: This is discussed under 'Naming'.
- Concurrency transparency: Where several users have to operate concurrently on shared data, NetWare provides a degree of concurrency transparency. No interference between users will be experienced as long as the data is read-only. If the data is read-write, NetWare provides locking mechanisms that will prevent unwarranted interference, but this is not transparent. NetWare also provides the Transaction Tracking System[13,18] (TTS) which provides safeguards against inconsistent data during updates. The data can be "rolled back" to its prior state if any problems arise during update.
- Replication transparency: All the VM database servers and the replication of applications on file servers are transparent to the user. Other replicated data, not directly part of the VM, will not be transparent to the user.

- Failure transparency: If a file server fails on the network, it is a total failure as far as all the users created on that file server is concerned. However, if a VM database server fails, then users will be transparently connected to another VM database server to gain access to information necessary for the operation of the VM environment. As far as files are concerned, the TTS provides a degree of failure transparency, because even if a PC or a server crashes during operations on files or groups of files, they will remain in a consistent state.
- Scaling transparency: This is discussed under 'Scalability'.
- Migration and performance transparency are closely connected to distributed processing and are thus not discussed here.

3.4 File Service

One of the major features of the NetWare operating system is the transparent access to files offered to requesting PCs. This transparency allows programs that were designed for a single computer environment to readily operate in a network. The NetWare file system resident on a file server is presented to a user on an MS-DOS computer as an MS-DOS file system in much the same way as MS-DOS files on a floppy disk in a local disk drive are presented to the user. This is achieved by mapping files on the file server to the local MS-DOS file system. Drive letters 'F' and upwards are used for this purpose.

The local MS-DOS file system is responsible for the accessing of the local disk drive. Any request for data is channelled to a device driver that performs the actual disk access. If the data is on a network file server, then the NetWare shell (which is equivalent to the local device driver in this case in that it accepts requests and returns data) is used. The MS-DOS file system keeps track of which device driver handles which disk drive. Network requests are handled by 'NETx.COM' which decides where the data is located, forms the appropriate request packets and then sends the data to 'IPX.COM' to manage the actual sending and receiving from the relevant file server.

To create the illusion of a local file system, the NetWare shell maintains information about drive letters whether local or network based and if network based, then which file server it refers to. The file servers also keep information about all the PCs connected to them. What appears to be a disk drive to a PC could in fact be just a portion of a disk drive on the file server. A request for data on a network drive (on the PC), is translated to a request on the file server for the specified data, with the help of all the information maintained by both the PC and the file server on that drive.

As an example, when a user requires a file called 'F:WP\THESIS.DOC', the NetWare shell (from the information it has on the drive letter 'F'), sends the request to the associated file server, say, 'EG'. The file server checks the information it maintains on the PC making the request and 'sees' that the letter 'F' is mapped to 'SYS:HOME\E\L\ISAAC' on the file server's local drive. The path provided by the PC is then appended. A request is then made on the file server for 'SYS:HOME\E\L\ISAAC\WP\THESIS.DOC'.

3.5 Naming

File servers

The file servers on the network are given unique two letter names. This is the shortest name length[19] permitted by NetWare. The main reason for this is to reduce the number of characters that a user will have to type to login to a file server. In a single file server setting and in the original NetWare environment, the user would type[20]:

```
LOGIN USERNAME
```

Now the user has to type:

`LG SN/USERNAME` (where `SN` is the two letter server name).

The total number of characters that the user has to type remains the same in both environments, i.e. 14 characters (including the space) as shown above.

In the VM, there is no advantage in maintaining long server names. In the normal NetWare environment, servers are given names that users can remember so that they can login to the appropriate file server. It is therefore not helpful in that environment for file servers to have cryptic two letter codes. In the VM the user does not need to know the names of the other file servers, so a two letter code to differentiate the file server is sufficient. A two letter code allows for at least 676 (equals 26*26) unique file server names. This is more than sufficient for all the file servers on the VM. When informing users of their login names, they are told that their login name is SN/USERNAME. In this way, the average user is not even aware of the name of the file server he has an account in.

Users

Each user is created on only one file server in the network. The user's login name should be unique on that file server. When the user is informed about his login name however, he is told a small lie. The login name given to the user will be of the format: <server name>/<user name>, where he only has to supply <user name> if his PC is already attached to his home server. This arrangement of including the server name in the login name given to the user violates location transparency. If the user is moved to another file server, his given login name will have to change. But this transference of user accounts from file server to file server is expected to occur very rarely. The benefit of the arrangement however is that users can login to their file server and be authenticated from anywhere on the network. No 'middle man' is necessary to supply the information as to where the user has been defined. The <user name> portion of the login name does not have to be globally unique. Other existing NetWare applications (mail packages for example) can be used without any modification, because these applications will see a normal Novell network. As long as the user's account is not transferred to another file server, the user can always be accessed by his login name irrespective of where he is on the network.

Application batch files

As far as possible, batch files to start applications should be given unique names. All the file servers that have a particular application can have a batch file with the same name to start up that particular application, but different applications should have different batch file names to start them up. This ensures that if a user is attached to more than one server, he will not inadvertently start up the wrong application with that batch file name. Since the user starts up the application of choice through a menu, the name of the batch file is unimportant to the user. Even if the application changes location, the user is shielded from that change as long as he selects that application from the menu. Through a combination of the application database directory and the local PC menu, location transparency for applications is provided.

3.6 Scalability

The computer network can easily be extended as more computers become available (to the limits of the networking technology and configuration[21] used and to the limits provided for by NetWare). Software that runs on the PC which helps in the presentation of the VM environment can be obtained from a file server after logging in. This means that updates to this software can be made available to all PCs. In other words, apart from the PC software[19] required to enable that workstation to be part of a NetWare computer network, no other software (or hardware) need be present on the local PC.

All data and application files are stored on file servers. The only unique item on the workstation is its node address. This feature reduces labour requirements in extending the network. Adding a new PC to the network involves merely connecting it to the network and loading MS-DOS system files and two NetWare files.

Dividing the user population into groups according to access rights (i.e. the VM categories), helps create a scalable system because users are only created on one file

server, but still have access to all the file servers via the VM category. Thus the addition of a user to the system does not require much labour.

Updates to the VM database is easily made because it is a centralised database. The new copy of the VM database can also be remotely loaded onto the VM database servers. This feature also improves scalability as all management of the VM database is performed in one location.

3.7 Security

NetWare authentication schemes[13,17] are employed. The user is verified when logging on to his resident server. If he successfully logs in, he will belong to a VM group with certain access privileges (or restrictions) to applications. When the user selects an application from the menus presented to him, an attachment is made to the relevant server on which the application he selects resides. His VM group determines the name that the station software will use to attach him to the other server. The software also supplies the password. This password is unknown to the user and is greater than 125 characters in length. This password is derived from a character string obtained from the VM database and is then converted by a function within the PC software. This prevents unauthorised access to resources.

The servers in the system are physically secure and run server software that users can not modify. No user programs are executed on the servers. Although PCs are not secure, malicious actions by individuals on one PC will not affect users on other PCs.

3.8 Compatibility

All the VM programs are expected to execute on all the PCs without modification. All the PCs are assumed to have INTEL microprocessors and the programs have been written for the INTEL 8086 processor (which is about the lowest in the series). Higher processors are downward compatible, so they should be able to execute 8086 instruction code without any difficulty. Currently, the programs cannot run on a different hardware platform without extensive modification.

Software that communicates with the VM database server only needs protocol compatibility.

3.9 User Mobility

User Mobility is a feature supported by NetWare and hence by the VM. This is because the VM software just builds on top of the capabilities offered by NetWare. Any user can obtain system services from any PC on the network as well as gain access to their private files.

3.10 Station Mobility

System or network services are not dependent on the address of the station. This means that the PC can be moved to any locality on the network without affecting the access capabilities to the network. The PC simply has to be unplugged from one location and reconnected to the network at another location. The system administrators do not need to be informed when a PC is shifted. If the PC is moved to another sub-network, the only thing that will change is the network address of the PC. But this is assigned at boot time. This feature will also allow portable computers to be connected to the network.

3.11 Easy Application Access

The menus presented by the VM PC menu program removes the difficulty experienced by users in a multiple file server network in locating applications. When the user selects an application from the menu, a connection to a file server containing that application is made and a batch file that will start the application is run. This batch file prepares the MS-DOS environment and creates search mappings that will permit the application to locate files it needs for its operation. In this way the actual access is also made easy.

Information about applications is stored in the VM database. Thus any changes to location and type of applications need only be noted in the VM database. This then becomes available to all users through the PC software. The user does not even have

to know where the application is or how to access it and is shielded from the changes made to the application's location.

3.12 VM Database Administrator's Job

This job is unique to the VM environment. The VM database administrator creates and manages the VM database and co-ordinates with the file server administrators to determine the VM groups and access rights for applications. He also sets the base passwords with which the various categories of users in the VM can login to other file servers. The menu program running on the PC will expand the base password to a 125 character string that will enable login to another file server when this is required.

The VM database administrator also selects the PCs on the network that will act as VM database server machines. He invokes the VM database server program on those machines, and from time to time when the VM database is changed, he remotely reloads a new copy of the database on the VM database server machines. There is not a great deal of work involved in this job, so the overall VM system manager can perform these duties as well.

3.13 File Server Administrator's Job

The file server administrator's job does not change much from the single file server NetWare environment. The file server administrator creates or deletes users on each file server as normally done[17] in NetWare systems. He has to copy the necessary programs needed for the operation of the VM on to the file server. He also runs a program that creates special groups and special users that will allow for VM access to that server by users normally resident on other servers. Then the restrictions imposed on these special users are set as with any other user or group. All applications should be run from batch files. The contents of the batch files follow a standard format (described in Appendix B2). When an application is added or deleted from the file server, the VM database administrator has to be informed to make the necessary changes to the VM database.

Before placing an application on a file server, the server administrators should contact the VM database administrator to see whether the application has already been installed in a file server somewhere else on the VM. The VM database administrator and the file server administrators should together determine what application access rights should be given to the VM groups. In this way, the problem of users being presented with different access rights depending upon the file server the PC software has chosen for them, is avoided. If a server administrator wants certain users on his file server to have additional rights, he can create a group on this file server that contains these users and gives the new group all the rights that he feels these users should have.

If the sub-network is large, it can have one file server administrator. If it is small, then the overall VM system manager can administer the file servers on it. Each user is defined on only one file server and belongs to only one access category. Generally, system administration is not very difficult, as the more numerous and physically dispersed PCs can be ignored leaving the relatively small number of file servers to require attention.

3.14 Local Sub-network Control

A degree of autonomy at the local sub-network level is a necessity in the UNZA situation. Each school would like a say in how its resources are being used. To allow for this, the local system manager (or whoever is administering the resources at that level) sets the trustee directories and queues of the special users accordingly[17]. In this way, all local users can have greater access to applications and other resources than users who are resident on other servers.

CHAPTER FOUR

VM DESIGN

The previous chapter described the application access technique and gave a summary of the operation of the VM. This chapter looks at some of the details of the VM design.

One of the most important components of the VM is the VM database. This chapter mentions the files that constitute the VM database and how it is created. The reason for having the VM database server is explained and its operation and the services provided are briefly presented.

Another important step in the operation of the VM is the preparation of the file servers. Some of the more significant aspects of how this is done are mentioned.

The users of the VM only see three programs: the login and logout programs and the VM menu program. These programs are briefly described in this chapter.

4.1 The VM Database

The technique for accessing applications is built around the VM database. Information necessary for the operation of the VM is served out to requesting PCs by the VM database server. There was a choice between implementing the VM database on the file server (using the file server's bindery) and on a dedicated stand-alone machine. A dedicated stand-alone machine was preferred for the following reasons:

1. If a file server is servicing a printer then the operation of the file server is slowed down and this will affect a search of the bindery.
2. The bindery of the file server is limited to any combination of up to 65000 objects and properties[22]. A large bindery is slower to access than a small bindery and will affect the overall performance of the network.
3. A dedicated stand-alone machine can be used as a central database limited only by its available storage space.
4. All information is available at a single location. This reduces the time involved in obtaining desired information as it avoids having to scan the binderies of various file servers.

The VM database files are still primarily located on a file server, but are loaded into the memory of the VM database server machine on start-up.

4.1.1 VM Database Files

The VM database files at present are:

- MENU.DBF. This file contains all the information on the menus that are to be presented to the requesting PC.
- AP_ACCES.DBF. This file contains all the information in regard to which VM group can access a particular application and the type of operating system and hardware needed to run the application. Only the access category is being used at the moment.
- AP_BATCH.DBF. This file contains information about the location of applications and the batch file that has to be run to start the application.
- AP_BATCH.NDX. This file is created by a program run by the VM database administrator (BATCH.EXE) that sorts the AP_BATCH.DBF file. The AP_BATCH.DBF is sorted firstly according to application ID (in ascending order) and then according to server name (also in ascending order). AP_BATCH.NDX is an index file that keeps an index of the position at which a particular application

occurs in the sorted AP_BATCH.DBF file, and the number of consecutive records after this one that has information about that application.

- USER.DBF. This file is not used at present. It is only provided for future use in the creation of a network wide user directory.
- USER.NDX. This file is an index to the USER.DBF file. It is not used at present.
- PASSWORD.DBF. This file contains all the information about the VM groups, the VM group login name for file servers and their base-password.

The structure of the VM database is elaborated in Appendix A1.

4.1.2 Creating and Maintaining the VM Database

A program called CDB.C was written in C specifically to create and maintain the VM database. CDB.C has mouse and windowing support. Considerable care went into making the program user friendly. Filling up the AP_BATCH.DBF file with application IDs is an activity that is prone to error. The possibility of error has been reduced by providing a menu from which the application ID can be selected. This menu has the NAME and APPID fields of the records in the AP_ACCES.DBF file. The tedious activity of filling up the MENU.DBF file has been made less burdensome by providing two menus from which all selections can be made. The first menu has information on the NAME and ID of applications and the second menu has information on the NAME and ID of menus.

4.2 VM Database Server

The VM database server acts as a 'middle man' whom the PCs can request data from, and who in turn services the request from information found in the VM database. If the PC software were to find the needed information from the VM database files itself, the middle man could be eliminated. This is an alternative technique, but the following should be considered when implementing it:

1. All users should have direct access to the VM database. The VM database can be replicated on all the file servers and every user given read-only access to the VM

database files on their home file server. Alternatively, all the VM database files can be located on one file server, and all the users should have read-only access to the directory containing those files on that file server. Then the VM PC menu program can read the various files and present the menus to the user and connect to various file servers on the user's behalf. The menu program will only have as much rights on the file server as has been assigned to the user, so a malicious action by a user will not affect everyone else.

2. Performance of the file server could affect the operation of the VM. The user will notice that the menus are being presented to him more slowly when the file server is being heavily used than when it is not.

3. Limited number of connections to file server. If the VM database files are only located on one file server, then the total number of connections to that file server can easily be reached if many users simultaneously try to access the database. NetWare v2.15 has a limit of 100 concurrent connections and NetWare v3.10 has a limit of 250 concurrent connections. This limit could also pose a problem to the VM database administrator during updates.

4. Updating the VM database could be difficult. As long as any user is accessing information from any of the VM database files, update of that file will be impossible. NetWare will indicate a "file in use" violation if an attempt to delete that file is made. This can lead to inconsistencies in the copies of the VM database held on all the file servers as well as prove inconveniencing to the VM database administrator. Users could keep some database files open for unusually long periods of time by 'pausing' the menu program just after it has opened the files. The same situation could arise if the menu program 'hangs' and the user leaves it in that state.

The VM database server is dedicated to serving requests from the PCs for data necessary for the operation of the VM. The contents of the VM database files are loaded into the memory of the server PC. In this way, the speed at which requests are

serviced is improved. There is also no problem with the number of connections because the PC uses a connection-less protocol to communicate with the database server. Since users do not have direct access to the database files, updates can easily be made. The disadvantage with the current arrangement is that a few extra PCs have to be used specifically as VM database server machines.

4.2.1 Hardware Requirements

Any IBM AT class PC with a network card can act as either the user's PC or the VM database server PC. More information on the requirements can be found in Appendix A1.

4.2.2 Operation

When the program is started up, it loads all the VM database files into memory from disk. After this stage, the disk is no longer accessed. Then the program advertises its presence on the network (and continues to do so every 60 seconds) and begins to listen for requests from PCs. All requests are placed in a queue by an event service routine and are serviced on a first-come-first-served basis. The server does not attempt to determine whether a particular request from a PC has already been serviced. (This is important because the PC and the server use the NetWare IPX protocol to communicate with each other — IPX is a connection-less or datagram protocol. Delivery of the response is not guaranteed.) This allows a PC to receive required information even if the initial server response was lost on the network. Information depending on the request type is sent to the requesting PC and then the next request is serviced. This process continues until the server is terminated. At this time the VM database server broadcasts a message telling all file servers and bridges on the network, to delete it from their internal lists.

Another feature of the VM database server is that it can receive a request from a PC to reload the VM database into memory. These requests are made by the VM database administrator, by using the 'LOADDB.EXE' program after altering the VM database in

some way. On receipt of this request, the program will stop listening to the PCs, service all remaining requests in the queue and then reload the database from the server indicated in the request. Afterwards, the server will resume the service offered to PCs.

4.2.3 Request and Response Buffers

All communication by PCs with the VM database server is through two buffers. The first buffer contains PC requests to the VM database server. The second buffer contains the response from the VM database server. The details of the two buffers are presented in Appendix A1.

4.2.4 Services

The services provided by the VM database server are briefly described below. More information about the available services can be obtained from Appendix A1.

1. *Get Password*: This will return to the requesting PC information about VM group names, VM group login names for file servers and VM group passwords for file servers. The VM database server obtains this information from the password.dbf file.
2. *Get Menu*: This will return the menu specified by the requested menu ID (equals record number in the MENU.DBF file starting from zero).
3. *Get Application*: This will return information about all the file servers that contain the requested application and the batch files that have to be run to start the application.
4. *Load VM Database*: This will load the VM database from the specified file server. The requesting PC will be notified whether this operation has been successful or not. Reasons for failure could be: none of the files were loaded, or only a partial loading was achieved. If no files were loaded, the server resumes with the old database. If a partial loading was made, the VM database server terminates.

4.2.5 Reliability

All PCs should be able to locate a VM database server. This is essential to the operation of the VM. To improve reliability, the entire VM database can be replicated on a few PCs through out the network. Since the VM database does not contain any information that changes at run time this is not a problem. If one VM database server goes down, PCs will still be able to contact another server and obtain the necessary information.

4.2.6 Extendibility

As other services become available, information about these services can be placed in the VM database. The VM database server program will have to include support for these services and then the program will have to be recompiled. Some changes might have to be made to the PC menu program as well.

4.3 Preparation of the File Servers

File servers are initially configured as any normal NetWare file server[19] and are given unique two letter names. All users are created and trustee rights[17] to directories are set as well. Application files are also placed on the file server's hard disk. The file server administrator however has to take a few additional steps (details given in Appendix B2). The directory 'SYS:PUBLIC\BIN' has to be created. All application batch files will be stored here. Files necessary for the operation of the VM will also be placed here.

4.3.1 Copy VM Programs

The VM programs have to be copied to specific directories on the file server for the operation of the VM. A batch file, called VM_COPY.BAT, that manages the copy operation is available. This batch file is presented in Appendix B2.

4.3.2 Create Special Groups for VM Access

A program, called VM_CREAT.EXE, that creates VM groups and a few special users (according to the VM group login name) and sets their passwords has been written. (Users who have not been created on a file server can login with the name of one of the special users because that is the VM group login name.)

The syntax is: VM_CREAT [<full path of location of password file>].

If the password file is not specified then the program will try to contact a database server for the information. The group name and the user name are obtained from the file (if the database server is contacted the file is password.dbf). The received password forms the base from which a password that is greater than 125 characters in length is generated. The length of the password and its encryption are for security reasons.

4.3.3 Set User Restrictions

All users that have been created on this file server are placed in one of the VM groups. Each user should only belong to one. The VM group they belong to determines their access category. Details of the restrictions imposed on the special users (i.e. those corresponding to a VM group login name) are available in Appendix B2.

4.3.4 Create Batch Files for Applications

Every application should be started from a batch file. A description of the format and an example of a batch file is given in Appendix B2. This format has to be strictly adhered to because the batch file prepares the MS-DOS environment and creates search mappings that will permit the application to locate files it needs for its operation. The batch file also has to register the file server in which the application is located in a temporary file called a Server Attachment Count (SAC) file. Each time an application on a file server other than the user's home file server is run, the attachment count to that file server in the SAC is incremented by one. Similarly, when quitting from an application, the batch file has to decrement the attachment count for that

server by one. When the attachment count for a file server reaches zero, that file server may be safely detached. Failure to follow the prescribed format for the batch file could result in the user having the following problems:

- the user may be detached from the file server where the application he is running is located when he requests for the VM menu. This problem normally occurs when the application is located on a file server other than the user's home file server, and has not been registered in the SAC file. Consequently, when the VM menu program terminates, it will detach all file servers that have an attachment count of zero including the one containing the application of interest. Additionally, if the application needs to access the hard disk of the server in which it is located, it will not be able to do this and so from that moment onwards would not be able to function properly.
- the application or the files necessary for the correct operation of the application can not be located and hence the application can not be invoked or made to execute properly.

4.4 Login Program — LG.EXE

A special login program to be used in place of NetWare's login program has been developed. This program calls the login utility supplied by Novell. Upon successful login to a file server, a SAC file is created in the location pointed to by the MS-DOS 'TEMP' environmental variable. The name of the SAC file is composed of the letters SAC, a full stop (.), and an extension — which is the connection number to the home file server. So, if the connection number of a PC to a user's home file server is '7', then the name of the SAC file for that user would be 'SAC.7'. If a user has logged into the same file server twice, he will have two SAC files corresponding to the two file server connection numbers of the two PCs he has logged in from.

The SAC file contains the following information:

1. the user's home file server's name.
2. the user's login name.
3. the user's login time.
4. the drive attachment count array.
5. the segment address of the parent's Program Segment Prefix[23] (PSP).
6. the number of unterminated applications on each file server that the user has selected.

Each time an application found on a particular file server is started, the attachment count to that file server in the SAC file is incremented by one. When the application ends, the attachment count is decremented by one. In this way, it is possible to determine when a connection to a file server has to be terminated.

The SAC file is created by the login program because it is easy to determine the home file server at this stage. Afterwards, a program could change the home file server, and then it would be difficult to locate information about the user. When the login program ends, the SAC file only contains the home file server's name, the user name, the user's login time, the drive attachment count array, and the segment address of the parent's PSP.

The first three items in the SAC file are for the purpose of establishing whether the SAC file is the correct one for the session. The home file server provides information on who is logged into that file server on that connection. The user name and the user's login time from the file server are compared with the information in the SAC file before any of the other information is used.

The parent's PSP is stored so that all the environmental blocks from the parent onwards can be located later. It is possible to have many copies of the MS-DOS command processor 'COMMAND.COM' in memory, and each copy would have an environmental block associated with it. Every program that is run under a particular

copy of the 'COMMAND.COM' would normally inherit the MS-DOS environment at that level. In this way, it is possible (and sometimes happens) that the user will be operating several levels away from the original parent's environment. Some programs need to set an environmental variable to a value in all the environmental blocks from the current environment up to the original parent's environment. The normal method in MS-DOS PCs, to chain backwards from the current PSP up to the original PSP, does not work. The NetWare redirector shell (NETx.COM) seems to change the field in the current PSP that points to its parent. Consequently, it is only possible to go back one level using the standard method. To solve this problem, another method using the original parent's PSP and the MS-DOS Memory Control Blocks[23] (MCBs) has been devised. This method needs the location of the original parent's PSP to work and it seems best to place this in the SAC file.

The structure of the SAC file and (more details of the LG program) is given in Appendix C11.

4.5 The PC Software —VM_MENU.EXE

The main PC program called 'VM_MENU.EXE' is usually run from within a batch file called 'VM.BAT' which should be available to every user upon successful login. The content of 'VM.BAT' is given in Appendix B2.

VM_MENU.EXE is not a TSR program. When the program terminates it is unloaded from memory. In this way it does not occupy any memory at all on the PC after terminating. This could always be important when working with MS-DOS PCs. This program terminates in three ways; on fatal error, on selection of an application or if the user chooses to 'QUIT' the program. Each time the user wishes to see the menu, he types VM<ENTER>. If necessary the batch file can be modified so that whenever an application ends, the user is returned to the menu.

4.5.1 Operations

User Information and Locating Database Servers

All information about the user, the file servers attached to, and the number of attachments to the file servers is recorded from the SAC file on start. All known database servers up to a maximum of four are located. The first database server to be located is set as the preferred server. All communication will be with this server. If this server fails, then communication will be established with the next server on the list and so on until all the servers on the list have been checked.

User Access Category

The user's access category is then determined. To do this, the bindery in the user's home server is scanned for all the groups the user belongs to. This information is compared with the information that the database server supplies out of the password.dbf file. If a match for the VM group is found, the user's access category is set to 2^x (where x is the record number of the password.dbf file where a match was found, and $0 \leq x <$ total number of access groups). From the same record, the user's VM login name, and password-base is found. (The user's actual VM password is determined from the password-base.)

Managing Menus

A request for menu with ID = 0 is made to the database server. This is always the main menu and is presented to the user on receipt. The user can select the desired option with either the keyboard or the mouse. If the selection is another menu, the required ID is communicated to the database server. In this way, a list of all selected menus is kept on the PC up to 20 levels deep. The user can go back up the levels of the menus by pressing the 'ESC' key on the keyboard. Backtracks do not require communication with the database server. However, any selection made at that level is still fetched from the database server, even if it is an option that has previously been selected at that level. In this way, the menus are managed until an application is selected from a menu.

Selecting the File Server with the Application

When an application has been selected, the database server provides information about all file servers that have the application and the access categories that can use the application. If the user has the correct access rights, the VM_MENU program then decides which file server to attach to, based on the following considerations in order of priority:

- Is the application present on the home server? Select it as the preferred file server for that application.
- Is the application on any other file servers that the PC has already attached to? Connect to the first one that has been found.
- Is the application on any active file server? Connect to the first one that has been found.

The Environmental Variable — VMPROG

When VM_MENU ends it is unloaded from the PC memory. The information to start the program must accordingly be stored in a safe place. The following is therefore done:

- A search mapping is made to the directory on the file server containing the batch file that will start the application.
- The name of the batch file (and the drive on which it is located) is stored in the environmental variable VMPROG. This environmental variable is set in all the environmental blocks up to the parent environment.

Example: Suppose a batch file called 'APP.BAT' found on a file server called 'EX' and stored in the directory 'EX/SYS:PUBLIC/BIN' will start the application that the user has selected from the menu. Firstly a search mapping is made to this directory if it does not already exist, e.g.' W: = EX/SYS:PUBLIC/BIN'. Then the environmental variable VMPROG is set to 'W:APP.BAT'.

Ending

If the program ends due to an error, the error level is set to 1. If the program ends because the user chooses to quit the menu system, the error level is set to 3. If an application has been selected, the error level is set to 0.

On normal ending, the program detaches from servers to which a connection need no longer be maintained, and current information is placed in the SAC file. The display is also cleaned up and files are closed.

4.6 Logout Program - LGO.EXE

This program deletes the SAC file created for the current session and then logs out the user to terminate the session.

CHAPTER FIVE

TESTING AND DISCUSSION

This chapter provides more information on the VM through the tests performed on it and through the reactions of ordinary users, file server administrators, VM database administrators and developers of applications for the VM. Results are given in sections 5.1 to 5.8 and are discussed separately at the end of each of those sections. The remaining sections (5.9 onwards) discuss other aspects of the VM, some of which is based on sections 5.1 to 5.8, and others from the preceding chapters.

The LAN configuration of Figure 5.1 was used during the testing stage. The hardware components in Figure 5.1 are listed in Appendix A2.

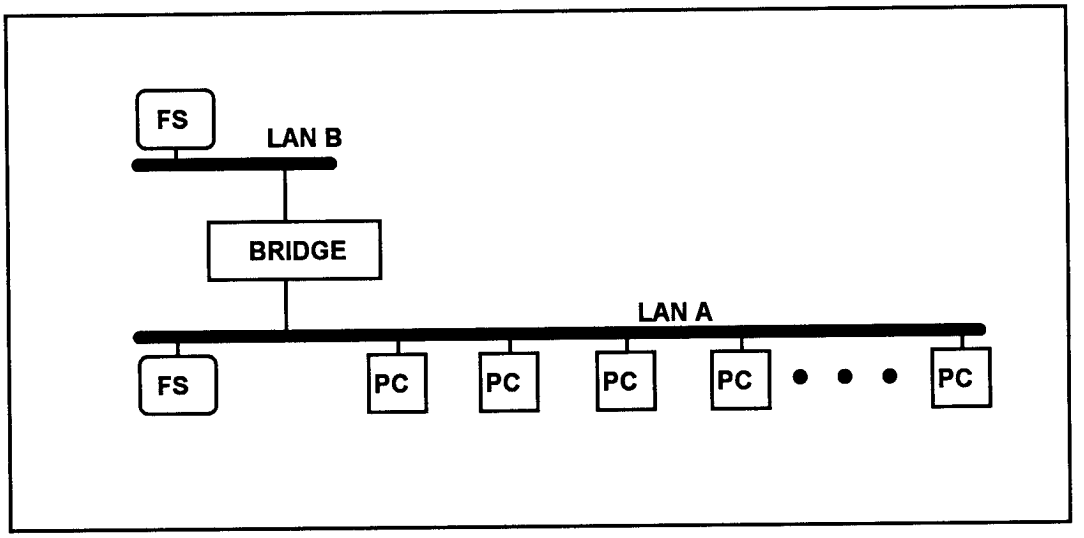


Figure 5.1: LAN configuration during tests

5.1 General Operation

For three weeks, all the students and staff who are registered users of the School of Engineering network used the VM system. During this period the VM database server was running on the non-dedicated bridge. All problems that occurred were reported. To start the system, all users were given access to the user manual (which is also presented in Appendix B3). Some applications were also transferred to the file server on LAN B.

The size of the VM database used in the test is given in Table 5.1:

FILE NAME	SIZE [BYTES]
AP_ACCES.DBF	864
AP_BATCH.DBF	648
AP_BATCH.NDX	144
PASSWORD.DBF	1422
MENU.DBF	6240
TOTAL	**9318**

Table 5.1: The Size of the VM Database Used in the Tests

The number of items in each category of the test VM is shown in Table 5.2.

CATEGORY	NUMBER
USERS	246
APPLICATIONS	36
MENUS	13

Table 5.2: The Number of Items in Each Category in the Test VM

Three VM groups were defined. These are: LECTURER, STUDENT and MANAGER. The group STUDENT had the majority of users. The group manager had only one user. Out of the 246 users, 243 were defined on the file server on LAN A and 33 out of the 36 applications were also located there. One of the applications located at the file server on LAN B was used quite extensively. One of the users defined on the file server in LAN B also tested all the applications presented by the VM menu program.

The following general observations were noted:

Applications that included explicit drive specifications in their executable files gave some problems (e.g. error messages such as 'CAN NOT FIND FILE' and other problems related to not being able to locate a needed file) to users whose home file server is different from the one on which the application is found. Even under periods of heavy demand, the maximum number of items in the VM database server queue was only one. (On one occasion, there were two items in the queue to be serviced. This information was collected by maintaining a small table in the memory of the VM database server PC that was updated each time before servicing a request. This table was printed to the screen whenever the VM database server was terminated. The table was monitored during the three week observation period. The code for doing this has not been included in the VM database server program, DB.C, listed in Appendix C3.)

Applications whose executable files have explicit drive specifications are thoroughly unsuited for use on a multiple file server LAN (even if the VM is not used and the network is treated simply as a normal Novell LAN) because the drive letter might refer to a file server's hard disk other than the intended one. Such applications should not be placed in the VM, or else should be available only for users who have an account on the file server where that application is located. If the application's set-up can be modified at run-time by a batch file using the MS-DOS SET[16] command, then a program called VMSET (written for the VM and described in Appendix C7) can

redeem the situation by mapping the server and volume name to a drive letter and then performing an operation similar to the SET command.

5.2 User's Reaction

The vast majority of the users on the Engineering network do not have any experience in working with multiple file servers in a Novell environment. It is therefore not possible to obtain a comprehensive comparison of user reaction to the VM system as opposed to the normal NetWare multiple file server environment. Some users who do have the relevant experience claim that the VM is indeed easier and more convenient to use than the normal Novell environment with multiple file servers.

In the normal Novell environment, the user has to perform the following steps to start an application:

1. Determine which file server has the application required.
2. Login to the appropriate file server.
3. Determine how to start the application.
4. Logout.

If the user is already logged in to a file server and requires an application which is not present in the current file server (to manipulate data present in the current file server for example), he would have to perform the following additional steps:

1. Determine which file server has the application required.
2. Attach to the appropriate file server.
3. Map a drive letter to a directory on the attached file server.
4. Determine how to start the application.
5. Logout from the attached file server.

With the VM system in operation, the user has only the following three steps:

1. Login.
2. Run VM and select the application of choice from the menu.
3. Logout to terminate that session.

From the reduction in the number of steps performed by the user and the information the user needs to know to start an application, it can be stated (intuitively) that the users will find the VM more convenient. Most users are unaware of the existence of multiple file servers and only know of the VM simply as a menu system. The users who know where the applications are to be found are unable to access them in any other way than through the VM menu program. The menu selection is made using either the cursor keys or the mouse and in this way it is similar to many other menu programs. User reaction from staff and students in the School of Engineering toward the menu program has been favourable.

Two users enquired why they were not returned to the menu whenever the application they had selected had terminated. This was a difference from the NetWare menu program for the single file server environment that they were used to. These two users were satisfied with an explanation of the possible advantages of the current scheme. Most users when asked whether or not they preferred to be returned to the menu indicated that they did not have any preference at all. Some users did not want to be returned to the menu on termination of an application. It was discovered that the majority of these users used the VM to load the environment for specific compilers or they used the MS-DOS command line extensively.

From the information presented above (in this section), it can be stated that the users find the VM menu program to be a convenient and easy way to start applications. Nothing can be conclusively stated from the user's reaction towards not being returned to the menu program after the termination of an application. The batch file that starts the VM menu program can be modified if necessary, so that it will return the user to the menu after the termination of a selected application.

5.3 VM Database Server Response Times

To test response time (from the user's point of view) of the VM database server, a few programs were written (two of these programs are in Appendix A2). A brief description of the programs follows:

1. TEST.C. This program continuously makes requests to the VM database server for the main menu, and on receipt of the menu prints a dot on the screen of the PC from which the request is made. TEST retries up to a maximum of two times if it does not receive a response within the time-out value specified in the program. For each retry, it prints an asterix ('*') at the bottom of the screen. If the program still fails to get a response from the server, it will print the capital letter 'X' on the screen in the position it should have printed a dot if it had received a response. After printing either a dot or 'X', the test program repeats the request for the main menu.
2. TEST800.C. The same as TEST.C except that only 800 dots are printed on the screen and the time taken to accomplish this is measured.
3. TEST20K.C. This program measured the total time taken to receive 20 000 successful responses from the VM database server. The number of retries and failed attempts are also recorded. This process is repeated 10 times to obtain 10 sets of information totalling 200 000 successful responses. TEST20K does not print anything to the screen during each measurement set. It does however, print to the screen after each set of measurements and the same information is also recorded in a data file on the test computer's hard disk.

During the test there were no other users on the network.

Intuitively, we expect that the response time seen by the user will consist of the following components:

1. Processing time of the user's PC.
2. Processing time of the VM database server.

3. Network transport time.

The processing time of both the user's PC and the VM database server, should be improved by having faster PCs. Since the tests are being conducted on an Ethernet[7], it is difficult to determine the network transport time. As the number of PCs on the network that are transmitting information increases, we expect that the network transport time will increase[24] as well because of increased probability of collisions, busy periods, etc.

The factors affecting the response time from the VM database server (as seen by the user) which are examined in sections 5.3.1 to 5.3.3 are:

1. Speed of the user's PC.
2. Speed of the VM database server PC.
3. Queue size set on the VM database server.
4. Time-out value set on the program running on the user's PC.
5. Number of PCs trying to access the VM database server.
6. Effect of placing the VM database server on another LAN.

5.3.1 Effect of the Speed of the VM Database Server and the User's PC on Response Time

The purpose of this test is to see what effect the speed of the PCs that are operating as a VM database server and as the PC running the menu program has on the response time the user perceives when choosing an item on the menu.

For this test, the VM database server was running on a 16 MHz, 80286 PC (Olivetti M 290 S) on LAN A. The maximum value of the request queue size on the VM database server was 25. The user's PC was tested separately so that only one PC would make demands on the database server. The PCs in Table 5.3 were tested both as a requesting PC and as a VM database server. All tested PCs were on LAN A. The Norton Computing Index, a benchmark to measure PC performance relative to the IBM XT (from Norton Utilities), is also included in the table.

PC	Norton Computing Index
50 MHz Victor	102.1
25 MHz	13.2
16 MHz Olivetti M290S	8.1
10 MHz IBM PS/2 model 30	5.6
8 MHz Olivetti M24	1.9

Table 5.3: The PCs Used in the Test

The TEST800 program was used to time 800 successful responses from the VM database server to requests made by the user's PC for different database servers and PCs. The results from a user's point of view are shown in Figure 5.2. There was only one 8 MHz PC, so it could not be used simultaneously as a server and as a user's PC.

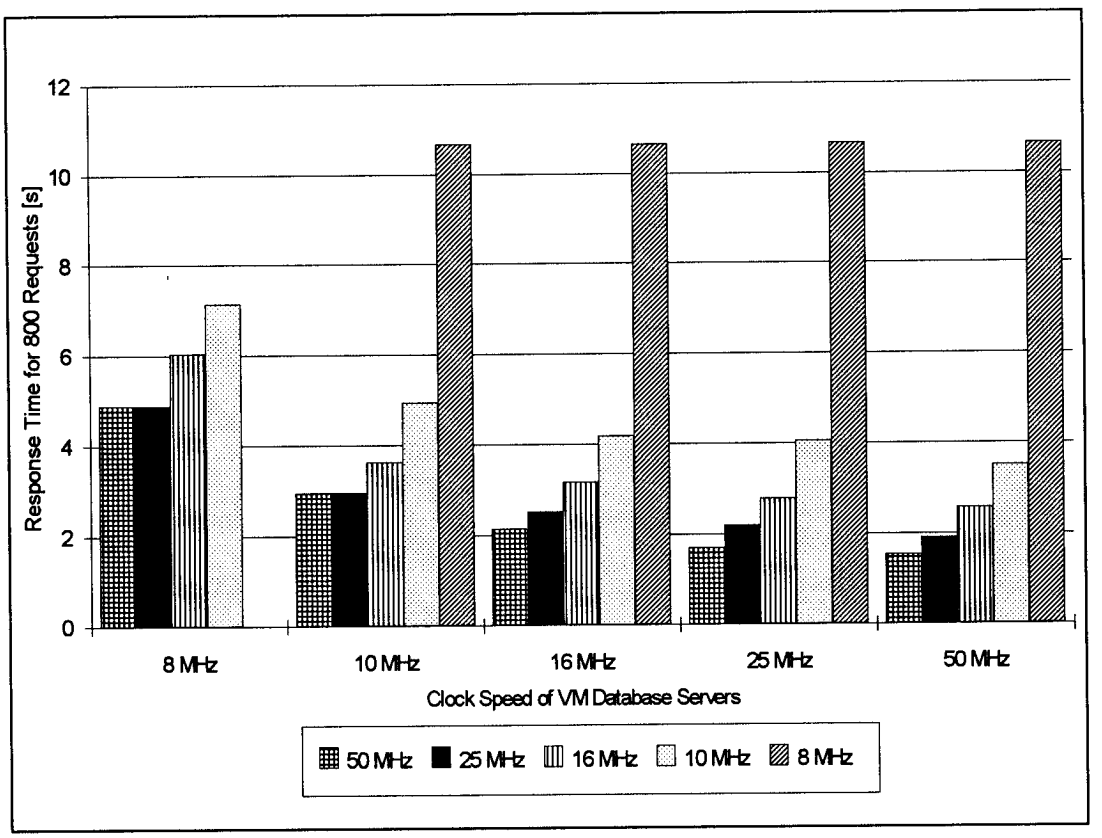


Figure 5.2: Effect of PC Speed on Response Time

Generally, the faster the PC running the VM database server program, the faster the response time the user will see. Also, the faster the PC running the menu program, the faster the response time the user will see. When the VM database server program runs on the slower PCs (8 MHz and 10 MHz PCs), the response time the user sees on the faster PCs (50 MHz and 25 MHz) will be the same. In this situation, the VM database server is probably the bottleneck in the performance. The faster PCs have to wait for a response from the VM database server. However, if the user's PC is too slow, then it becomes the bottleneck in the performance. This is inferred from Figure 5.2. Irrespective of the speed of the VM database server shown, the response time seen on the 8 MHz PC is the same. It can be said that the response from the server is waiting for the slow PC. The slowest response shown in the figure is approximately 9 times slower than the fastest response shown, but is still less than 0.05 s per request on average. This means the speed of the PC (used by either the user or the VM database server) will not noticeably affect the performance of the VM menu program from the perspective of a human being.

5.3.2 Other Factors Affecting Response Time

The purpose of this test is to see what effect changing some of the system parameters and the loading on the VM database server would have on response time.

TEST20K was used to measure response time (and some other useful information) for various combinations of time-out values on the test PC and queue sizes on the VM database server. The loading on the system was increased by having other PCs run the TEST program (also using the tested time-out values) concurrently with the PC running TEST20K. Only the PCs on LAN A were used for the test. The results are presented in Figures 5.3 to 5.6.

Please note that in this section: To = time-out value, q = maximum queue size, DB = VM database server PC, TWO DB = two DBs, T = average time taken for a PC to receive a response from a DB, R = average number of retries required for a PC to receive a response from a DB, N = number of PCs requesting information from DBs.

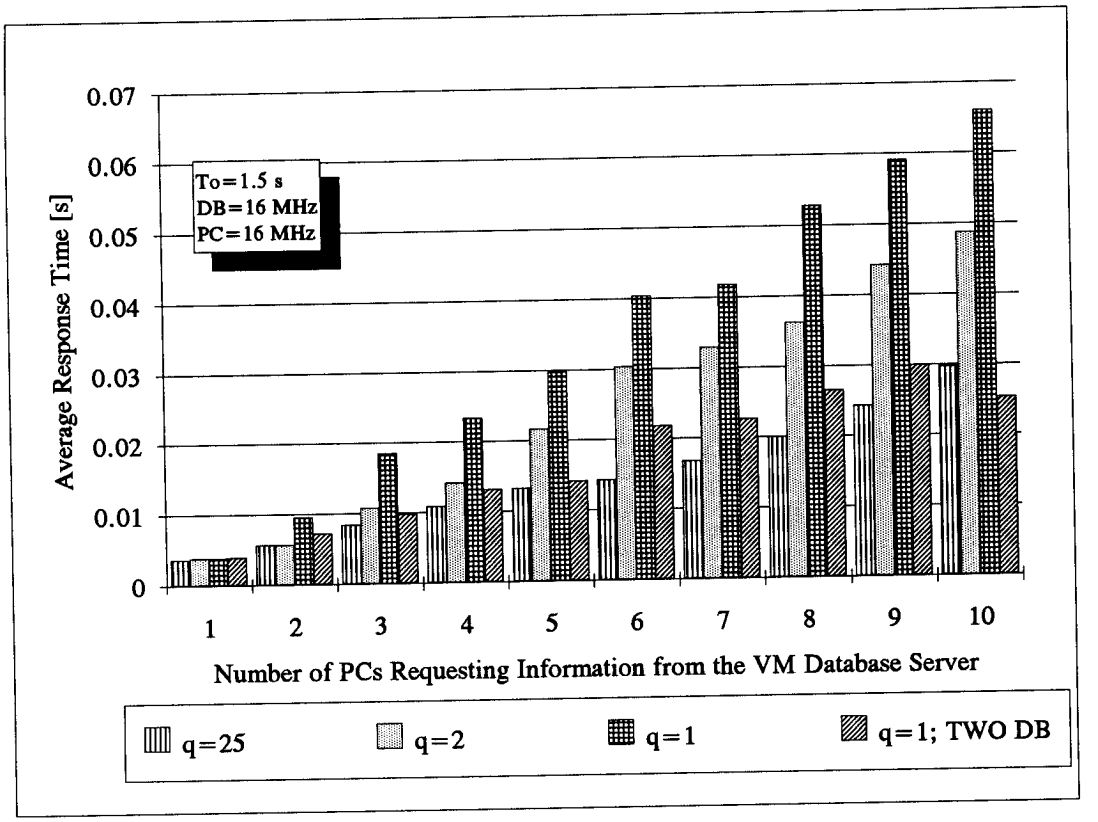


Figure 5.3: Effect of Varying Queue Size on Response Time

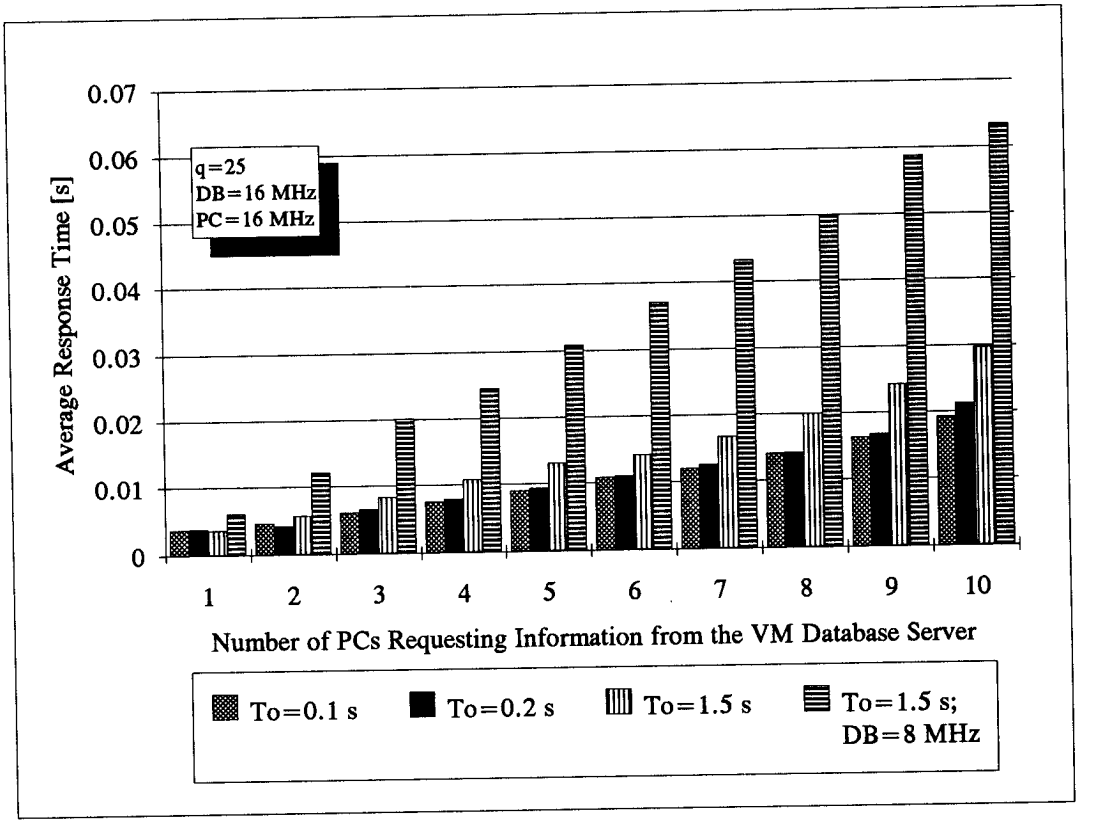


Figure 5.4: Effect of Varying Time-out Value on Response Time

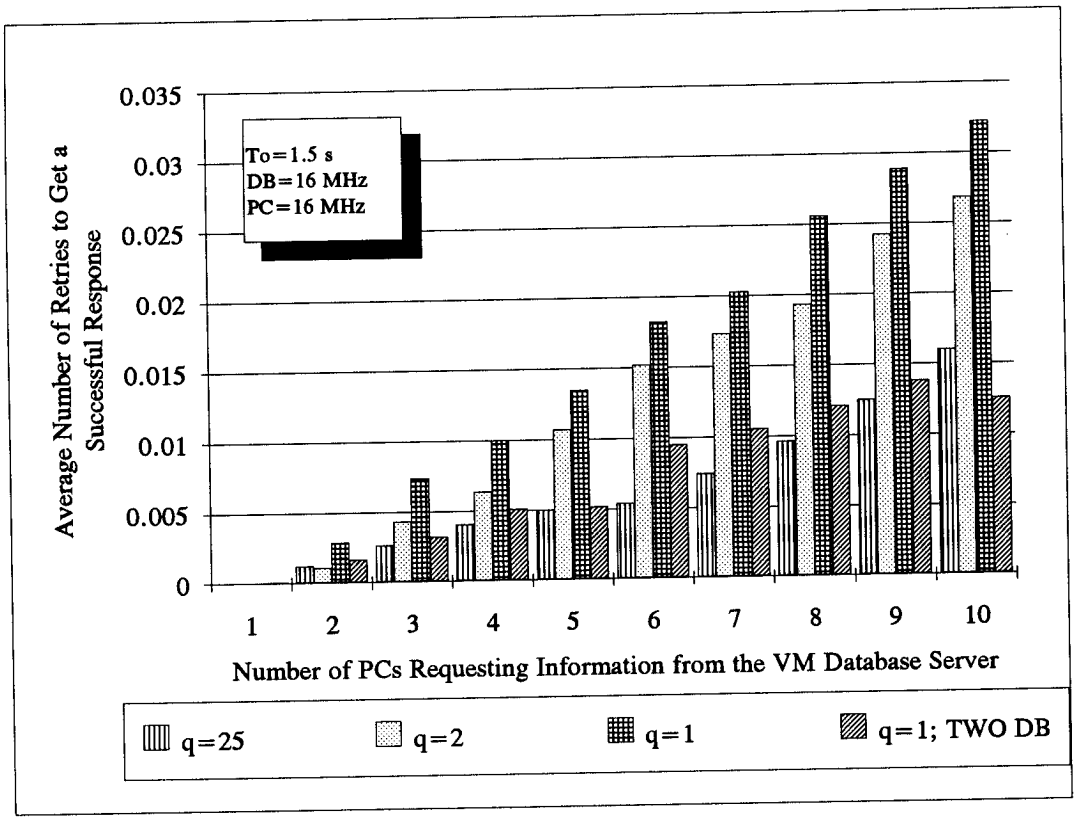


Figure 5.5: Effect of Varying Queue Size on Number of Retries

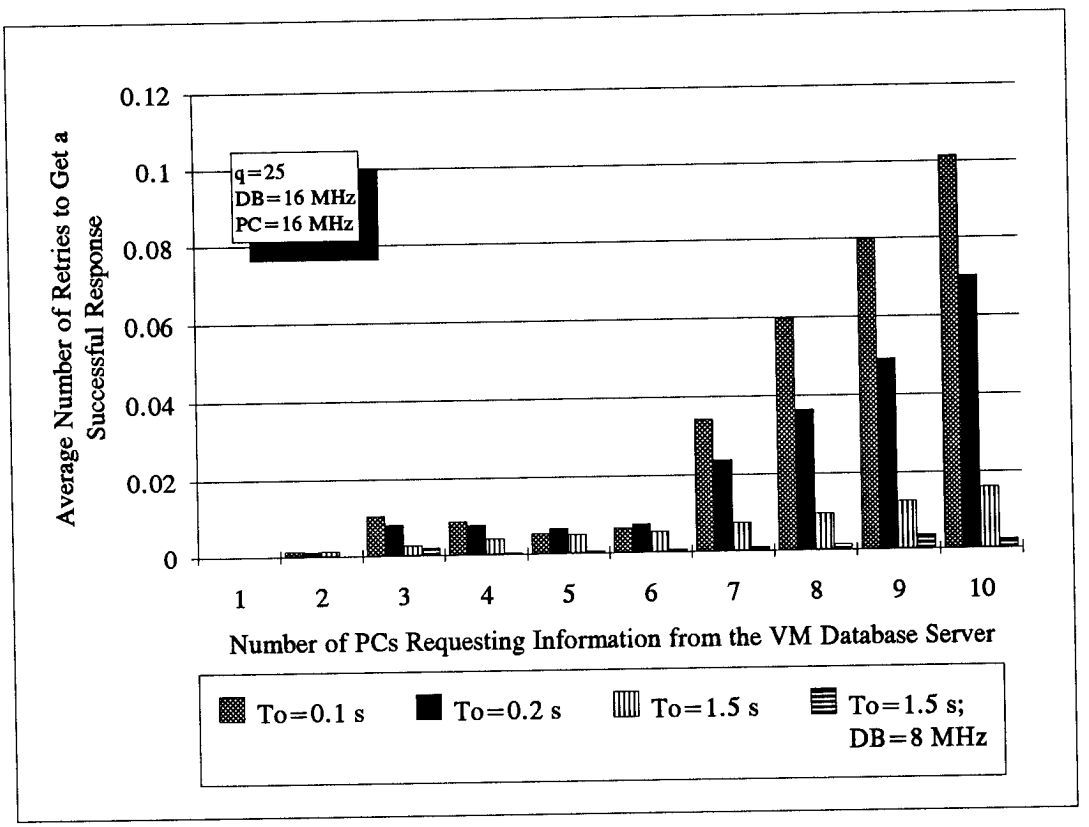


Figure 5.6: Effect of Varying Time-out Value on Number of Retries