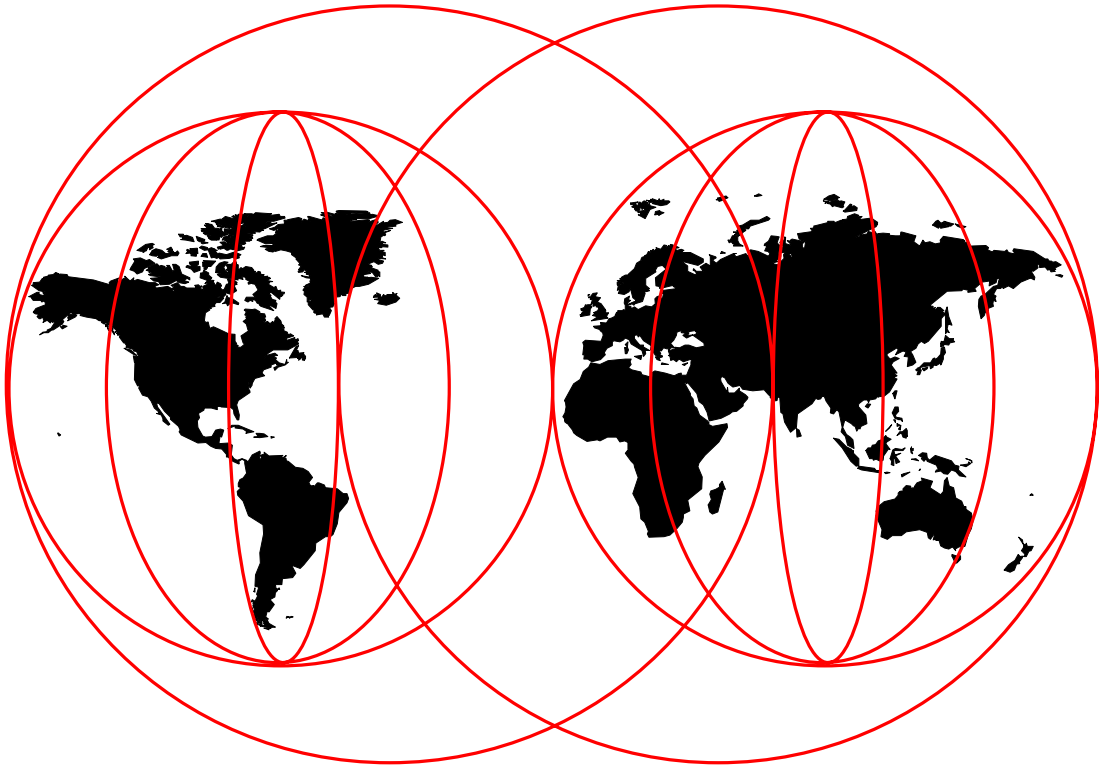
Consolidating UNIX Systems onto OS/390

Stephen Turner Ron Argent Michel Jan Frederic Mora
Yves Tognali



International Technical Support Organization

http://www.redbooks.ibm.com

SG24-2090-00

International Technical Support Organization

SG24-2090-00

Consolidating UNIX Systems onto OS/390

May 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix D, “Special Notices” on page 203.

First Edition (May 1998)

This edition applies to S/390 Parallel Enterprise Servers Generation 3 and later, running with OS/390 Version 1 Release 3 and later.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
522 South Road
Poughkeepsie, New York 12601-5400

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

This redbook describes the IBM S/390 Enterprise Server as a target for the consolidation of applications currently running on multiple distributed UNIX servers.

The objectives of this book are:

To show how server consolidation is economically justified in many situations.

To position S/390 within the spectrum of consolidation offerings.

To show that S/390 should be the platform of choice for selected environments.

To provide some practical guidance in the planning and execution of a project to consolidate multiple UNIX servers onto OS/390.

This redbook helps you position the S/390 among the possible target systems for server consolidation. It also offers guidance for planning and management of the project, and discusses some issues to be considered in the migration of the UNIX application to OS/390 UNIX Services.

How This Redbook Is Organized

The book is divided into two parts:

Part I discusses the business and technical advantages of UNIX server consolidation.

Part II provides guidance for persons performing such a consolidation to an OS/390 environment.

Part 1 consists of:

Chapter 1

Contains an introduction to the subject of UNIX server consolidation.

Chapter 2

Discusses the options that might be considered for a consolidated system and introduces the possible benefits for each of several consolidation scenarios.

Chapter 3

Contains a discussion of the particular benefits of a OS/390 solution. Key architectural features of S/390 and OS/390 are introduced together with their advantages for an organization consolidating UNIX application servers.

Chapter 4

Discusses the different performance characteristics of UNIX and S/390 systems. Issues of system sizing are introduced. Some suggestions on possible approaches to sizing the consolidated system are offered.

Chapter 5

Describes an actual customer experience in consolidating a single application from several separate UNIX systems (with different parts of the application running on different servers) to a single S/390 processor running OS/390.

Part 2 consists of:

Chapter 6

Introduces some aspects of project management and education requirements for a migration to a S/390 consolidated system.

Chapter 7

Discusses some implications on performance when porting applications from UNIX to OS/390 systems.

Chapter 8

Discusses the operational differences between typical UNIX systems and IBM's S/390 with OS/390. Recognition of these differences are critical to the success of the consolidation.

Chapter 9

Considers some differences in programming languages between typical UNIX and OS/390 UNIX services and which should be considered when migrating to OS/390.

Chapter 10

Discusses the role of OS/390 as a communication server, and some of the differences from traditional UNIX Web servers.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Poughkeepsie Center.

Stephen Turner is a Consulting System Strategist at the International Technical Support Organization, Poughkeepsie. Center. He writes extensively and teaches IBM classes worldwide on all areas of S/390 positioning. Before joining the ITSO in 1997, Stephen worked in IBM Australia for 29 years as a S/390 hardware and software Systems Engineer.

Ron Argent is a Systems Engineer in the S/390 Business. Based in the United Kingdom, Ron has over 20 years of experience in the IT industry. His current role focuses on server consolidation, as well as the development of core applications in the enterprise. Ron has produced other material on these subjects.

Michel Jan is currently a Project Manager in the S/390 New Technology Center in Montpellier, France. He has 10 years of extensive experience in Industry Applications and ERP projects implementation.

Frederic Mora is a software engineer in the S/390 New Technology Center in Montpellier, France. He has nine years of experience in the UNIX field and has written a book about the Internet. He also teaches C, TCP/IP and UNIX.

Yves Tognali is a Systems Engineer in France. He has 10 years of experience in the OS/390 and MVS field. His areas of expertise include Parallel Sysplex. He has written extensively on MVS and ESCON.

The authors would appreciate your feedback on this book. Ideas for future editions would also be useful to us. Please send your comments to the e-mail address:

`sjturner@us.ibm.com`

or to the standard redbook e-mail address given below.

We thank the following people for their invaluable contributions to this project:

Al Preston	IBM S/390 Division
Joe Temple	IBM S/390 Division
Pete Driever	IBM S/390 Division
Marina Greenstein	IBM SMPO, White Plains

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 213 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

For Internet users	`http://www.redbooks.ibm.com/`
For IBM Intranet users	`http://w3.itso.ibm.com/`

- Send us a note at the following address:

  `redbook@us.ibm.com`

Part 1. Why Should UNIX Servers be Consolidated?

Chapter 1. Introduction

UNIX is considered by many businesses to be a premier application development platform, and there are many thousands of applications available across various industry segments. During the early 1990s, there was a widespread adoption of decentralized client/server architectures, based on small, low-cost PCs or UNIX workstations. The attraction was the apparent low cost of the base technologies coupled with the speed of introduction of new applications. However, this proved to be a trend with many hidden costs.

Instability, the high cost of downtime, the labor-intensive nature of distributed systems management, support costs and other factors called the economics of decentralized servers into question. Many studies[1] over the last few years have shown that, when all costs are taken into account, systems deployed on distributed PC and UNIX servers are between two and five times more expensive than systems deployed on large central servers. Organizations are also finding that the *fragmentation of data*, common in a distributed system architecture, is affecting their ability to have a single view of their business intelligence and inhibits their ability to implement new services quickly.

The rapid deployment of network computing has led to a proliferation of Web servers, and to a demand for access to existing production applications and data from the Web.

Organizations with more than a small number of servers are finding that the incremental support, operational and management costs of additional servers are becoming very significant.

For the sake of clarity, the following definitions are used throughout this book:

Distributed servers This refers to servers located near the end user.

Centralized servers This refers to the same distributed servers but which are centrally located.

Clustered servers This refers to servers that work in parallel to process applications.

Many enterprises have large numbers of distributed servers. Most of these enterprises are finding that supporting distributed servers is labor-intensive and that the quality of the service is less than adequate for mission-critical applications. A widespread trend is to centralize these servers into one site in an attempt to reduce costs and improve service to their end users.

Once the servers are centralized, the next logical step is to cluster the servers so that they can be operated and managed as a single entity (for example, using multiple nodes of an IBM RS/6000 SP, or multiple logical partitions of an IBM S/390). Workloads of smaller servers can be further consolidated onto larger servers in order to save on hardware and support costs. Dramatic reductions in hardware and support costs can be achieved in this manner.

---

1 For example, see *S/390: Server Consolidation with a Vengeance* , Meta Group Enterprise Data Center Strategies, File 658, July 3rd 1997, *Cost of Scalability: A Comparative Study of Mainframe and UNIX Server Installations*, International Technology Group, 1997, and *More Evidence on the Cost of Distributed Computing*, Gartner Group Research Note, September 4, 1997.

The final stage of consolidation might be to more tightly integrate multiple applications onto a single system (such as OS/390 on a single S/390, or multiple S/390s in a cluster--a Parallel Sysplex).

1.1 Preliminary Discussion

Distributed systems, where many servers are located in user departments, are very common today. Because in many cases the financing for the application was provided by the user department, the server platform (vendor, server size and configuration) was often also selected by the user department. Thus there may be many different combinations of server hardware (vendor, system model and so on) and software (version of UNIX, tools and so on) in a distributed network.

Having many different platforms to manage increases the complexity of managing situations where the time to react is an important factor.

Reducing the number of suppliers and systems can introduce some direct savings related to the hardware, software and the number of people required to manage Information Technology (IT) within the enterprise.

Today there can be a difference in terms of system management when the customer has applications on a mainframe (for example, a S/390 Enterprise Server), or on UNIX servers, or both. A customer with a S/390 installed generally has built up an experienced, dedicated team of people to operate and manage the system. This team is generally composed of specialists in the different products used on the mainframe server. This may also be true where UNIX servers are located in the same site as the mainframe. In that situation there may be some persons dedicated to the management of *all* the servers in the same location.

During the late 1980s and early 1990s there was a proliferation of distributed server applications, where many servers were spread throughout the enterprise. This distribution of servers was done based on the department or entity that needed such a resource for a specific application. The result was that the management of the servers was done by the department, and by non-experienced people according to their will or availability (rather than by any existing IT department, where such skills may have already existed). This created a lot of environments where it was difficult to handle hardware problems, software problems, change management, and so on.

Some additional consequences of such a situation are:

- The backup of enterprise-critical data may be difficult, out of date, or non-existent.
- The validity and integrity of enterprise data may be compromised.
- There is an inability to handle recovery or disaster situations.

Later in this redbook, we discuss how the S/390 architecture, together with the OS/390 operating system, has solutions to these problems if the distributed servers are consolidated into a single S/390 server, or into a cluster of S/390 servers.

Classic OS/390 strengths: OS/390 is probably the most robust operating system in use today. From its initial genesis as OS/360 in 1965, it has evolved from a batch-oriented system to a complex and sophisticated operating system capable of handling efficiently and concurrently *all* types of modern applications.

These enhancements have been implemented without impacting its prime objective of providing unsurpassed application availability of service to users and applications, while maintaining binary-code compatibility for over 30 years.

OS/390, in addition to its ability to provide unparalleled levels of service availability to thousands of online users, modern database applications and batch systems, supports the modern technologies demanded by the users of today's distributed systems, such as:

- Internet Web serving
- The high security and integrity required for electronic commerce
- Java applications
- Object-oriented technology
- Conformance to industry standards, such as UNIX95

All of these factors make the S/390 Enterprise Servers, together with OS/390, a competing platform for any enterprise contemplating consolidation of distributed UNIX servers onto a centralized platform.

Many such enterprises also have existing S/390 installations running enterprise-critical online and batch applications. These S/390 systems are frequently the repository of large amounts of operational data upon which the enterprise depends, and which is often replicated out onto the distributed servers. (Several industry consultants have estimated that 60-80% of government and business data in the developed economies of the world resides on S/390 systems.)

Access to these existing applications/data from new Internet-enabled applications is often a key driver for consolidating distributed UNIX servers into a central site.

Chapter 2. Server Consolidation - Benefits and Options

Organizations today are facing increasing demands for reliability, availability and serviceability for their mission-critical commercial UNIX applications. This demand for improved service is dramatically increasing the support costs for these applications.

2.1 Benefits from Consolidation of Servers

As mentioned in Chapter 1, “Introduction” on page 3, many organizations are finding that as the number of servers proliferates, the cost and operational complexity are also rapidly increasing. In many cases there are concerns whether multiple distributed servers can provide the application availability, hours of service, responsiveness and ability to grow with the requirements of the business. These characteristics are being increasingly demanded by business applications.

To reduce these costs, many customers are attempting to consolidate their servers into a more manageable central location. The three steps of server consolidation discussed in the first part of this book are:

1. Recentralizing servers
2. Merging workloads onto a single large server
3. Consolidating servers onto one system

Some recent customer examples of this trend are:

- A large USA company that provides services related to the New York Stock Exchange installs a new server for each new customer. They have found that application stability deteriorates whenever the average server utilization exceeds 50-60%. Since they have to accommodate workload peaks of up to twice the average, they install a new server whenever the average server utilization exceeds 30%. Each new server requires a backup machine (two servers) and three additional people to manage/operate.

  This customer is planning to consolidate many hundreds of UNIX servers to a small number of S/390 servers clustered in a Parallel Sysplex.

- A major Italian bank's Treasury Management Service, written in C++, was running on UNIX servers and PCs in LANS or on standalone PCs. Because the bank was growing, it needed increased stability and availability.

  It was decided that running the application on OS/390 UNIX System Services would enable:

  – Integration of the application with their core banking applications.
  – Better financial control of the Treasury Management business.
  – An increased number of client workstations (from approximately 8,000 today).

  With the help of an Italian software support organization and IBM business partner, the porting commenced in mid-1996. Application testing began in March 1997 and the first of 120 servers went into production under OS/390 on their S/390 in September 1997. The full consolidation was planned for completion by the end of 1997. See the following url for further details:

  `http://www.s390.ibm.com/products/oe/banco.html`

- An American telephone company is looking at moving a mobile phone billing application from multiple UNIX systems to OS/390.

  The driving forces for this migration are:

  – The application is currently large and complex and consists of both batch and online processing. Online is impacted when the batch processing does not complete on time, and when online systems compete for resources while the remaining batch processing completes. Extension of the billing application to other areas is expected to exceed the capacity of the current platform in 1998.
  – One large database had already been split in order to get the application to meet performance expectations, and another database is projected to require splitting soon. These could require major logical restructuring of the application.

  Areas where consolidation to a S/390 server with OS/390 could be beneficial to this customer are:

  – The application was originally architected to run on a single machine. Growth has led to its being split over a number of systems requiring a large volume of SQL*NET calls and NFS usage, which are both costly in terms of resource usage. Consolidation to a single S/390 server would eliminate this overhead, as well as accommodating the application growth.
  – The current platform has very limited I/O bandwidth. This impacts a number of application areas, for example, the length of time for backup and recovery processes. S/390 with OS/390 is renowned for its I/O handling capability and the speed of its backup/recovery utilities.
  – UNIX is considered to be not well-suited to a complex commercial workload consisting of both batch and interactive tasks. The existing facilities for job scheduling and priorities are insufficient for the application. One of OS/390's major strengths is the ability to efficiently and dynamically schedule online and batch processing according to the policies of the enterprise (see 3.3.3.2, “OS/390 Workload Manager” on page 45).

As we discuss in 2.4, “Consolidation Opportunities” on page 12, there are a number of possible scenarios that can be considered for consolidation.

Generally speaking, the drivers for server consolidation fall into two groups:

1. Improved service to customers or end users
   - Improved availability of service (fewer or no outages, and shorter outages should they occur)
   - Potential for improved service times to the end user due to a reduction of the time to communicate between clients and servers in single location
   - Opportunity for new services by enabling or improving interaction between different applications, or between new applications and data in existing systems
   - Web serving/Electronic commerce based on secure access to data in existing production systems
   - Improved security and integrity of data
   - Support for larger numbers of users accessing the same application by running on a larger server than may be cost justified in a distributed environment
2. Lower operational costs:

- Lower systems management costs
  - Network management
  - Configuration management
  - Problem and change management
  - Operational management (for both automated and manual operations)
  - Lower security administration costs
  - Single pool of skills in one location rather than many.
- Reduction or elimination of user department operational costs
- Possibility of reducing some software licenses (and hence costs)
- Possibility of reducing the number of systems, disk storage costs, elimination of maintenance charges and so on
- Potential for greater scalability on a large centralized server (for example, IBM RS/6000 SP or S/390 Enterprise Server)

Costs may be lowered further if multiple applications can be run on a single server and if the peak activity in any given workload occurs during periods of lower activity for the other workloads. In this case the server size may be much less than the sum of the sizes of the servers that it replaces (for example, see Figure 29 on page 70).

This last benefit may only be achievable if the operating system used is capable of dynamically balancing the workloads according to both business priorities and the resource constraints of the server (see 3.3.3.2, “OS/390 Workload Manager” on page 45).

2.1.1.1 Areas of Potential Savings

The following costs need to be examined to determine the feasibility of consolidating servers.

Server Costs: Who provides hands-on problem support for the remote servers: the support staff or a service contractor? If it is the support staff, how much time do they spend traveling, how long is the server down before they can service it, and how many times do they have to visit the server in a month? Supporting remote servers is considered to be at least two to four times more costly than centralized servers and the service provided is typically much poorer. If the service is done by a contractor, what is the cost of the contract? What happens when a remote server has a hard failure? In many environments this can be a huge productivity hit to the end users.

Backups: How are backups done? Typically with distributed servers, if backups are done at all, they are done locally on a small tape drive. Estimate the amount of time needed for backups and the cost of that labor. Remote backups are also error-prone and a bad backup can cause an even longer outage (anecdotal evidence suggests that as many as 75% of such backup tapes may be unusable for recovery). With central backups, the equipment and support costs are reduced, while the reliability of the service is increased.

Upgrades: How does the customer conduct software and hardware upgrades for remote servers? Do they use their own staff, who must travel to the remote site, or do they contract the service out? Upgrades to distributed servers are typically serial in nature. Upgrades for centralized servers can be grouped for efficiency.

Utilization: How much wasted hardware is there? Frequently distributed servers are under-utilized, sitting idle for long periods of time. With centralized servers, workloads can be merged to share the server. Distributed servers can be two to five times as costly as centralized servers based on hardware utilization. Many organizations with UNIX servers report that they can only operate their servers at 50-60% utilization and that both performance and reliability degrade at higher utilizations

Software Costs: Another area to examine is the cost of software licenses. This is usually a major expense for distributed servers. Typically, customers need to buy a license for each server. By consolidating workloads into larger servers, organizations can save money on reduced software licenses.

Security: Another requirement of distributed computing is security. Commercial database applications are mission-critical and typically store very valuable data. Often physical security is overlooked. What prevents an intruder from breaking into an office environment and dumping data to a diskette or a tape? What prevents someone from walking away with a hard disk or the whole server? Moving the servers to a central location with controlled access helps reduce the risk, as does consolidating them to a larger, more secure server such as an IBM S/390.

Network: It is imperative that customers install a robust network prior to consolidating servers to a central site. The network must be highly available with sufficient bandwidth to support the client/server traffic during peak periods. A poor quality network will make the centralized servers appear slow and unreliable.

2.2 Types of Consolidation

As mentioned in 2.1, “Benefits from Consolidation of Servers” on page 7, in this book we examine three possible stages of server consolidation:

1. Recentralizing servers
2. Merging workloads onto a single large server
3. Consolidating servers onto one system

These three stages of consolidation are shown in Figure 1 on page 11.

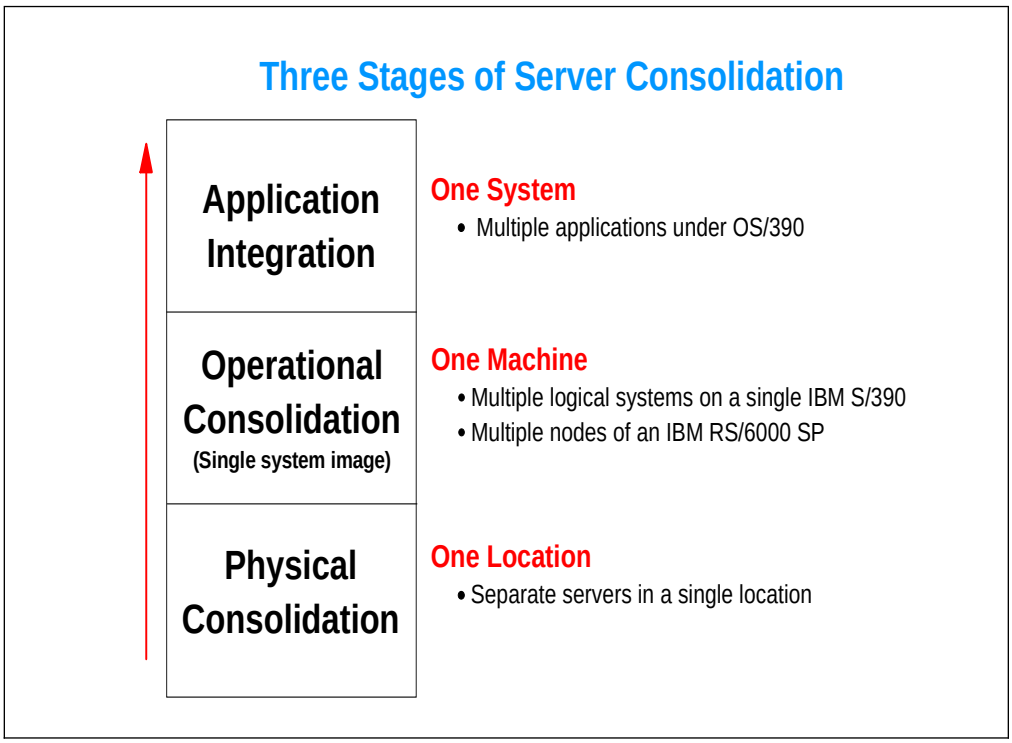


Figure 1. Three Stages of Server Consolidation

2.3 Types of Applications

We can categorize the applications that need to be considered into four broad groups:

1. Customized applications developed in-house or by tailoring general purpose applications from third party software vendors or from other organizations in the same industry
2. Packaged applications, for example, SAP R/3, J.D.Edwards, Walker, Peoplesoft, Lotus Notes or Domino, Baan and so on
3. LAN serving - for example, print servers, virtual disks or backup and archival
4. Web servers - with/without access to production systems and data

There can be business benefits for all four types of application when these are consolidated from multiple separate servers onto into a single location. Further benefits may be possible by then reducing the number of physical servers by running multiple logical systems on a reduced number of servers, or by integrating the applications into a smaller number of systems.

Three-Tiered Client/Server Applications: There are three components of an application: presentation (GUI), business logic, and data access. The GUI is separated from the business logic which is separated from the database. In this design, the simple and stable GUI is on the desktop, the business logic, is on an application server, and the data is on a database server. This shifts the maintenance problem from the desktop to the application server. It also shifts the network traffic from the desktop/data server interaction to the application server/data server connection. Three-tiered implementations favor the IBM S/390

with OS/390 or the IBM RS/6000 SP, as both the business logic and database can reside in a single physical system and can be managed by one interface. The extra reliability, improved performance, and systems management add to the justification.

Both the S/390 and the SP offer superior scalability for three-tiered client/server applications, with the ability to run up to 32 S/390 servers (in a Parallel Sysplex) or up to 512 nodes (in an RS/6000 SP), in one complex. Whether a particular application can benefit from this scalability opportunity depends on whether or not architectural constraints exist within the application itself.

2.4 Consolidation Opportunities

In this section we look at sample scenarios and the possible benefits in each.

Table 1 shows some possible consolidation opportunities and benefits for each server environment:

- Multiple copies of the same application on distributed systems
- One application with different parts of the application on different servers
- Multiple applications (one or more servers per application)
- Web servers (several servers per organization)
- LAN/print servers

Table 1 (Page 1 of 2). Consolidation Opportunities

Type of Server	**Benefit of Consolidation**
Multiple copies of the same application on distributed systems	• Lower operational costs • Better availability of service • Improved systems management • Single maintenance bill • Better version control • Better software distribution
One application with different parts of the application on different servers	• Lower operational costs • Better availability of service • Simpler, more reliable operation • Greater scalability • Improved performance • Improved systems management • Removal of communications between parts (lower overhead)
Multiple applications (one or more servers per application)	Opportunity for additional business due to the possibility of integrating applications. All benefits from above categories.
Web servers (several servers per organization)	• Lower operational costs • Improved ability to access operational data • Greater physical and logical security • Single source (one copy of data) • Improved systems management

Table 1 (Page 2 of 2). Consolidation Opportunities	
Type of Server	**Benefit of Consolidation**
LAN/print servers • Virtual disk • Print serving • Backup and archival	• Lower operational costs • Improved services to users • Improved availability of service • Improved ability to recover after an interruption to service • Improved ability to handle peak loads

2.4.1 Sample Scenarios

The following scenarios are discussed in this section:

1. Consolidating multiple instances of a single application
2. Consolidating an application which spans several servers
3. Consolidation of multiple applications running on one or more servers
4. Consolidating LAN servers

The first three unconsolidated scenarios are shown in Figure 2.

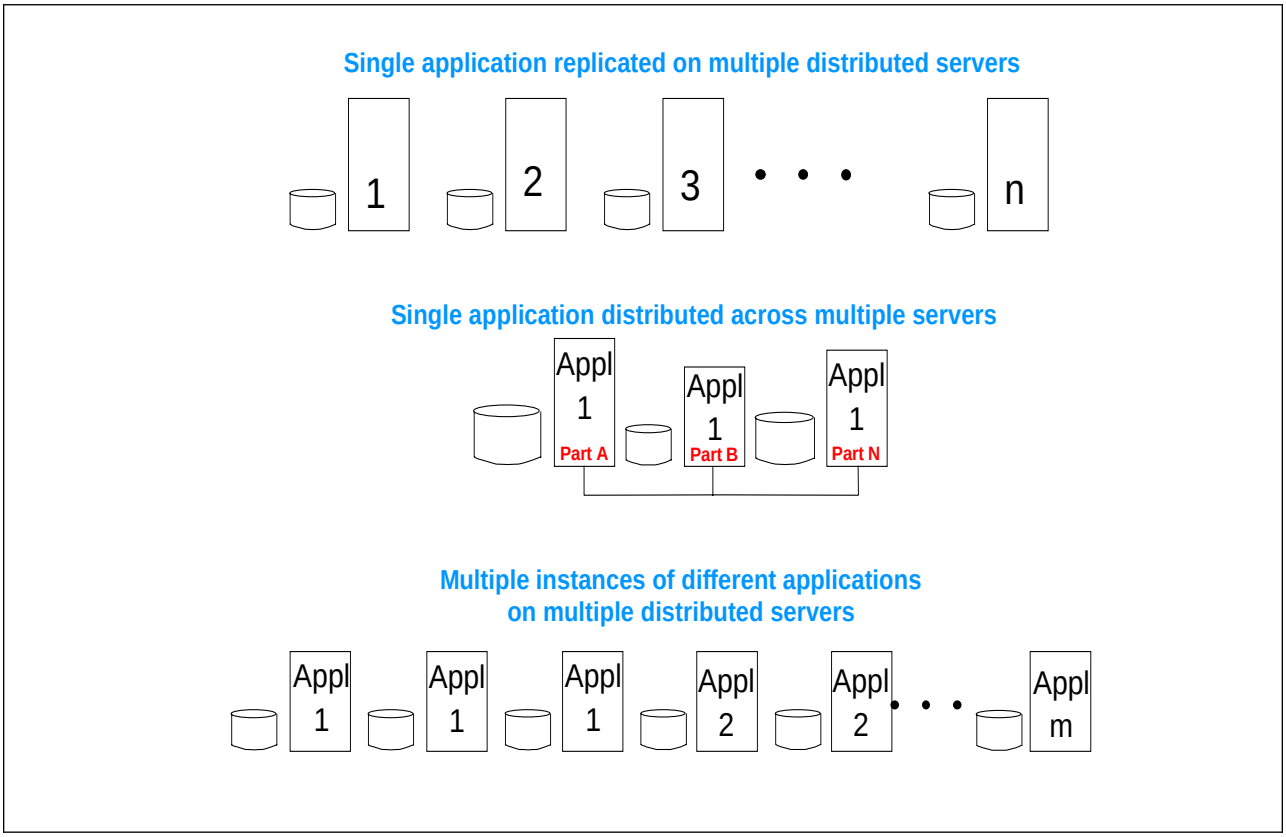


Figure 2. Sample Scenarios for Server Consolidation

The consolidation options considered are:

1. Moving the distributed servers into a centralized location.
2. Moving the distributed servers onto separate nodes of an IBM RS/6000 SP (or other clustered multi-node UNIX server). All nodes reside in one set of frames and can be operated from a single control workstation.

3. Moving the distributed servers onto separate logical partitions of a single S/390 processor. The multiple systems can be operated from one set of consoles.
4. Combining all occurrences of an application into a single instance of the application running on a single S/390 processor.

   Note: Where there are multiple applications, these could be combined into separate logical partitions, or run as separate applications under one copy of OS/390.

2.4.1.1 Multiple copies of a single application

Table 2 summarizes the potential benefits for each of these options for the consolidation of multiple servers, each running the same application.

An example of such an application might be that of an enterprise with departmental servers in distributed locations, each running a copy of a single application.

This might be either an in-house developed application or one purchased from a software vendor, and might operate independently during normal business hours, and then send business data, database updates, orders and so on overnight to some centralized site where enterprise-wide applications are run. The centralized system could be a large UNIX system, or very frequently an IBM S/390 processor with OS/390.

As seen from Table 2 and as supported by the examples in the succeeding section, there are business and financial benefits from all four consolidation options considered here.

Note that there is an increase in the number of benefits as we move through the stages of consolidation.

Table 2 (Page 1 of 2). Consolidation Benefits for Multiple Copies of a Single Application

Benefit	Centralized Location	Multi-node Cluster	Multiple Logical Partitions of a Single S/390	Applications Integrated on a Single S/390
Lower systems management costs	X	X	X	X
Improved systems management	X	X	X	X
Reduction or elimination of user department operational costs	X	X	X	X
Reduced software licenses		X	X	X
Improved availability of service to customers or end-users		X	X	X
Improved ability to recover after an interruption to service	X	X	X	X
Improved service times		X	X	X
Opportunity for new services			X	X
Improved ability to access operational data			X	X
Reduced operator costs		X	X	X
Reduced maintenance costs		X	X	X

Table 2 (Page 2 of 2). Consolidation Benefits for Multiple Copies of a Single Application				
Benefit	**Centralized Location**	**Multi-node Cluster**	**Multiple Logical Partitions of a Single S/390**	**Applications Integrated on a Single S/390**
Reduced number of systems to manage		X	X	X
Improved physical security	X	X	X	X
Improved logical security			X	X
Improved scalability			X	X

Figure 3 provides a pictorial view of these scenarios.

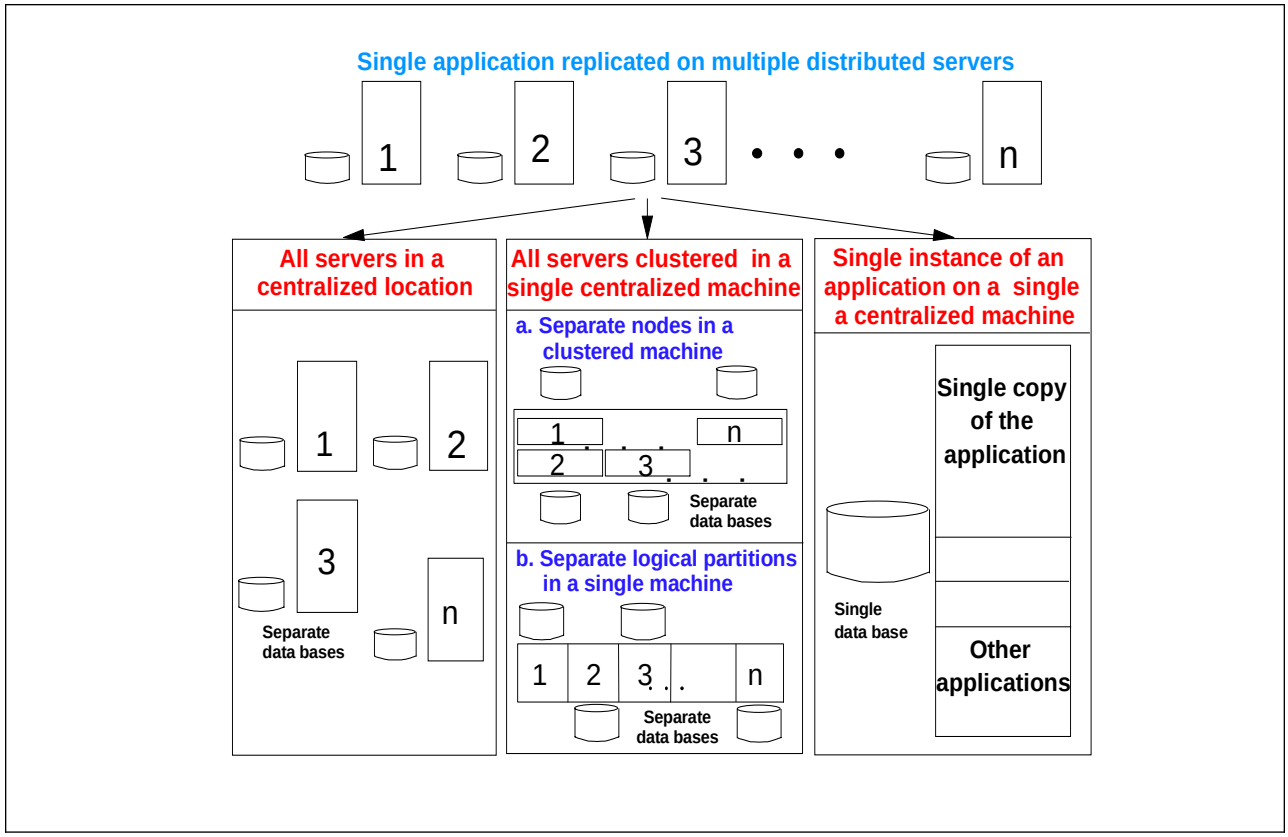


Figure 3. Single Application Replicated across Multiple Servers

2.4.1.2 Single application spanning multiple servers

Table 3 on page 16 summarizes the potential benefits for each of these options for the consolidation of multiple servers, each running a part of a single application.

Examples of such a consolidation are the three customers referenced in 2.1, “Benefits from Consolidation of Servers” on page 7, and the customer consolidation example given in Chapter 5, “Customer Consolidation Example” on page 83.

The types of benefit in this case are identical to those in the preceding example, except that the more integrated nature of the application means that communication

between the servers in the first centralization stage will probably be faster and more reliable leading to better, more responsive service for that stage.

Table 3. Consolidation Benefits for Single Application on Multiple Servers

Benefit	**Centralized Location**	**Multi-node Cluster**	**Multiple Logical Partitions of a Single S/390**	**Applications Integrated on a Single S/390**
Lower systems management costs	X	X	X	X
Improved systems management	X	X	X	X
Reduction or elimination of user department operational costs	X	X	X	X
Reduced software licenses		X	X	X
Improved availability of service to customers or end-users		X	X	X
Improved ability to recover after an interruption to service	X	X	X	X
Improved service times	X	X	X	X
Opportunity for new services			X	X
Improved ability to access operational data			X	X
Reduced operator costs		X	X	X
Reduced maintenance costs		X	X	X
Reduced number of systems to manage		X	X	X
Improved physical security	X	X	X	X
Improved logical security			X	X
Improved scalability			X	X

Figure 4 on page 17 provides a pictorial view of these scenarios.

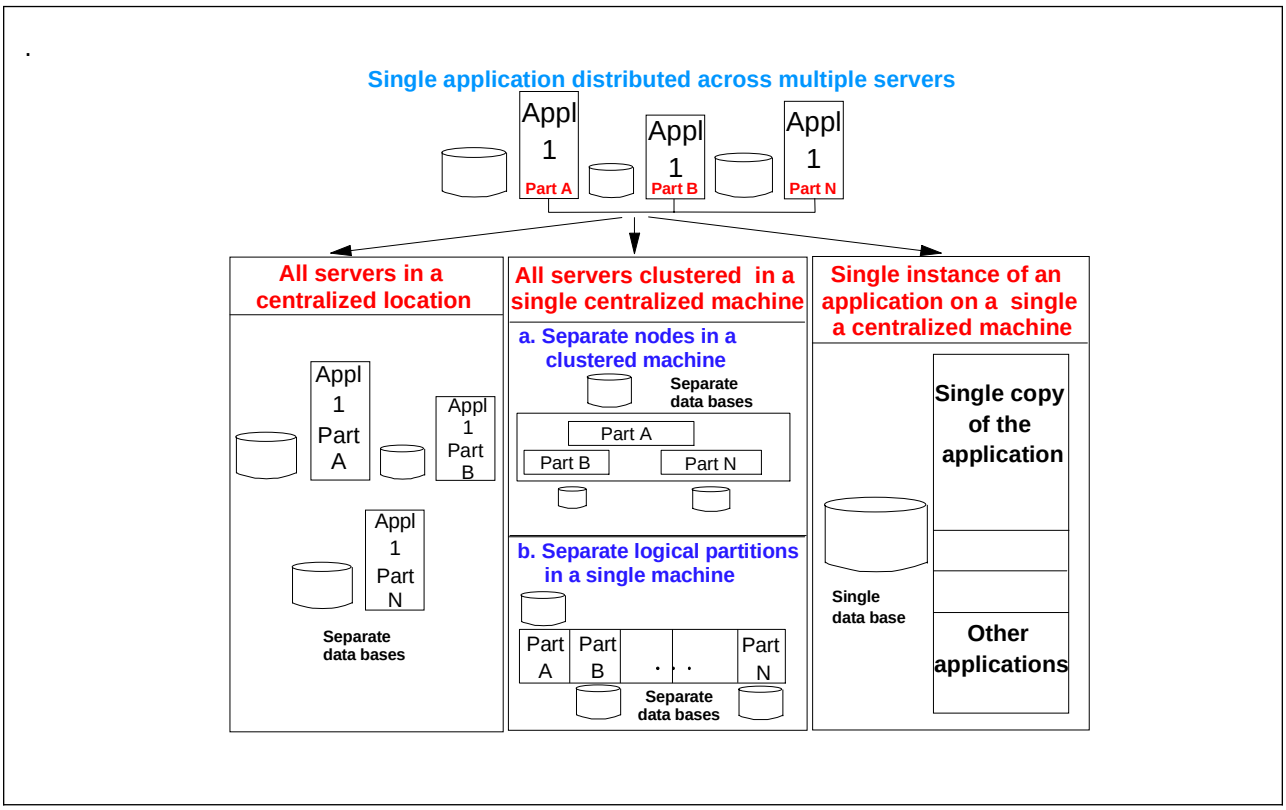


Figure 4. Single Application on Multiple Servers Consolidation Scenarios

2.4.1.3 Multiple applications

Table 4 summarizes the potential benefits for each of these options for the consolidation of multiple servers running multiple applications.

An example of this type of server consolidation might be that of multiple Web servers in an organization being consolidated.

In this case, the closer proximity of the servers could lead to easier communication between them resulting in improved response times, higher service availability of service and the opportunity for new services on the Web.

When there is an existing OS/390 system running mission-critical applications, consolidation to an OS/390 system may allow still further offerings since access to operational data in a timely, secure manner could lead to more tightly integrated Web-enabled electronic commerce offerings.

Table 4 (Page 1 of 2). Consolidation Benefits for Multiple Applications

Benefit	Centralized Location	Multi-node Cluster	Multiple Logical Partitions of a Single S/390	Applications Integrated on a Single S/390
Lower systems management costs	X	X	X	X
Improved systems management	X	X	X	X

Table 4 (Page 2 of 2). Consolidation Benefits for Multiple Applications

Benefit	Centralized Location	Multi-node Cluster	Multiple Logical Partitions of a Single S/390	Applications Integrated on a Single S/390
Reduction or elimination of user department operational costs	X	X	X	X
Reduced software licenses		X	X	X
Improved availability of service to customers or end-users	X	X	X	X
Improved ability to recover after an interruption to service	X	X	X	X
Improved service times	X	X	X	X
Opportunity for new services	X	X	X	X
Improved ability to access operational data			X	X
Reduced operator costs		X	X	X
Reduced maintenance costs	X	X	X	X
Reduced number of systems to manage		X	X	X
Improved physical security	X	X	X	X
Improved logical security			X	X
Improved scalability			X	X

Figure 5 on page 19 provides a pictorial view of these scenarios.

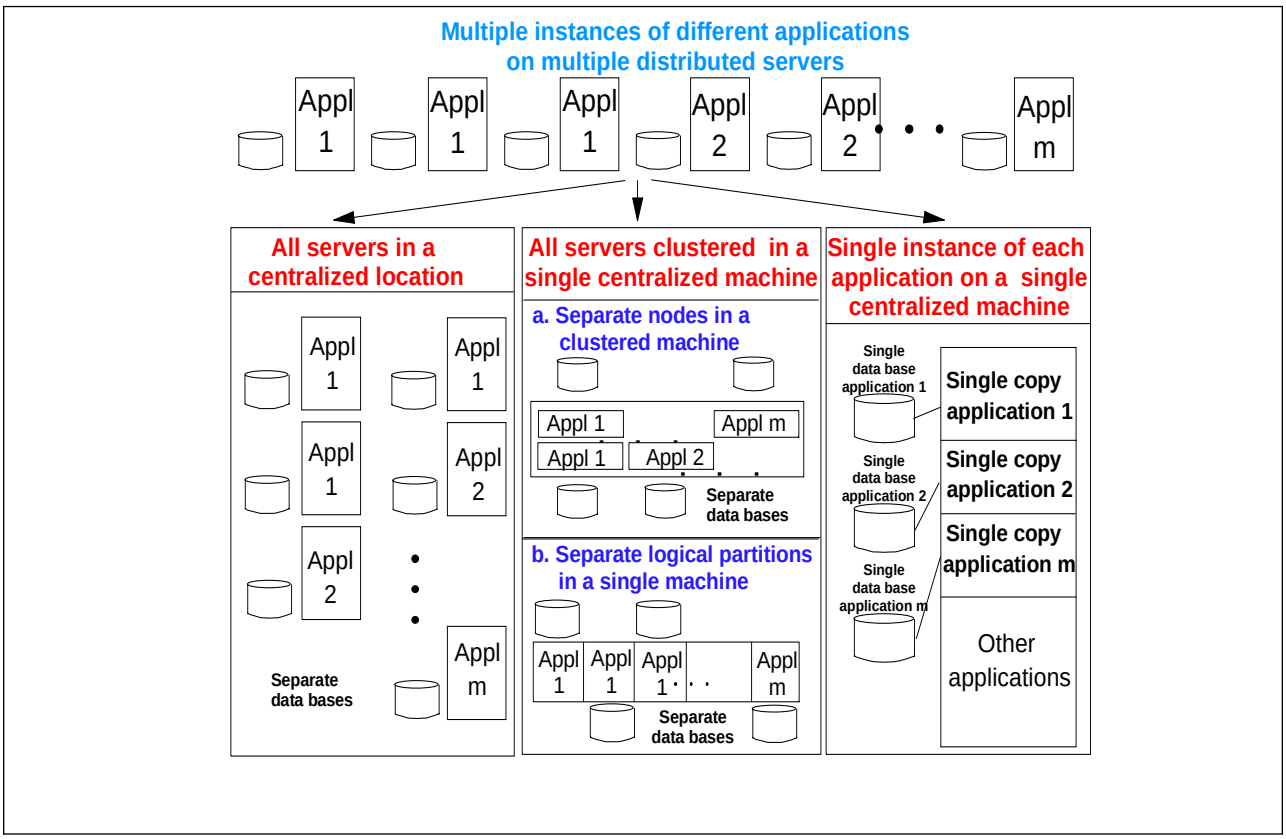


Figure 5. Multiple Applications Consolidated to a Centralized Location

2.4.1.4 LAN Server Consolidation

Table 5 on page 20 summarizes the potential benefits for each of these options.

Simple centralization offers the benefits associated with lower operational costs and the opportunity for the better systems management that can be possible with a pool of skilled staff in a single location.

However, further consolidation to a reduced number of higher function servers can improve the number and extent of services to the end user on a remote LAN.

These can include:

- Backup performed on a regular basis by data center personnel,
- The use of higher speed disks and tape devices for backup and recovery
- Consolidated administration of workstation data
- Enabling effective distribution of data to multiple LAN servers from a central location
- A higher degree of physical security
- Ability to quickly add additional disk capacity for use by LAN-based applications
- Enabling growth by using the storage capacity of the consolidated server to relieve the capacity constraints of workstation based servers
- Allowing end users to route print files to LAN or centralized host printers
- Allowing users of centralized applications to print files on LAN printers
- Enhancing systems management with support for centrally administering the LAN environment from the centralized site

- Reducing the cost and complexity of print environments by supporting enhanced print stream transforms and enhanced host-to-LAN print services

A fuller discussion on the use of an OS/390-based S/390 server for LAN consolidation can be found in 3.2, “OS/390 and LAN Data Integration” on page 33

Table 5. LAN Server Consolidation Benefits

Benefit	**Centralized Location**	**Multi-node Cluster**	**Multiple Logical Partitions of a Single S/390**	**Applications Integrated on a Single S/390**
Lower systems management costs	X	X	X	X
Improved systems management	X	X	X	X
Reduction or elimination of user department operational costs	X	X	X	X
Reduced software licenses		X	X	X
Improved service to customers or end-users		X	X	X
Improved availability of service	X	X	X	X
Improved ability to recover after an interruption to service	X	X	X	X
Improved service times		X	X	X
Reduced operator costs		X	X	X
Reduced maintenance costs	X	X	X	X
Reduced number of systems to manage		X	X	X
Improved physical security	X	X	X	X
Improved logical security			X	X
Improved scalability		X	X	X

Figure 6 on page 21 illustrates the consolidation of LAN servers onto a single OS/390 system.

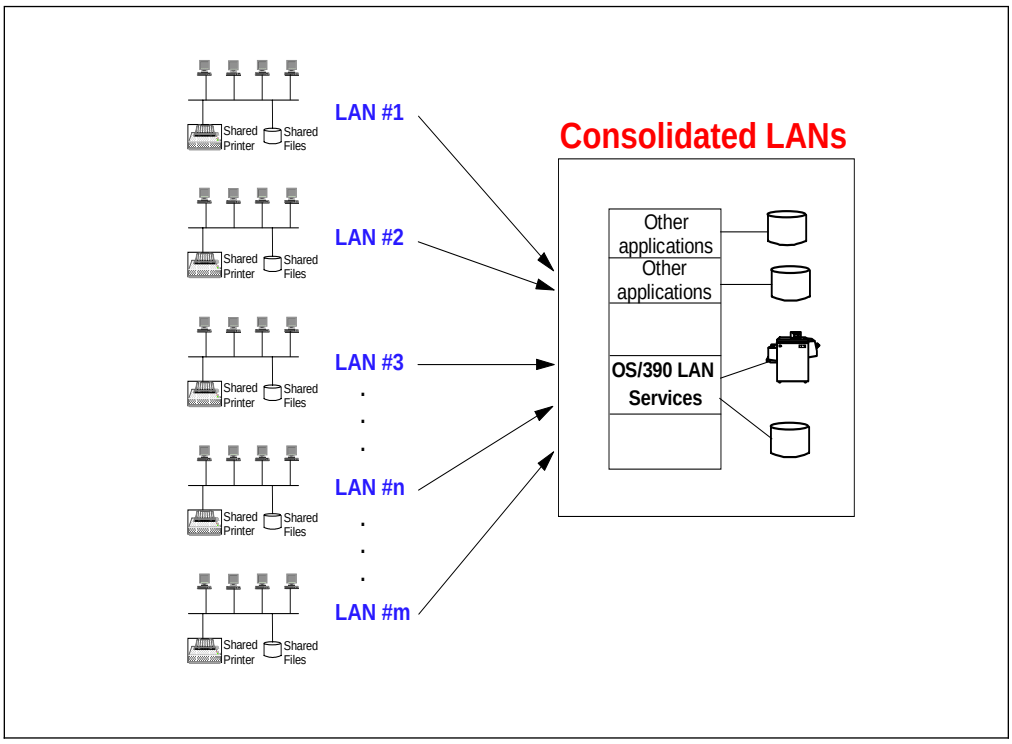


Figure 6. LAN Server Consolidation

2.4.2 Benefits of Each Stage of Consolidation

In the following sections we discuss the benefits that might be achieved from each of the stages of server consolidation:

- Centralizing Servers into One Location
- Consolidating Multiple Servers onto a Smaller Number of Servers
- Consolidating Multiple Systems onto One System
- Consolidating Servers onto One Operating System Image

2.4.2.1 Centralizing Servers into One Location

In 2.1.1.1, “Areas of Potential Savings” on page 9, we discussed the areas where savings from server consolidation from multiple distributed servers to a smaller number of centralized servers could be found.

These were:

- Server hardware costs
- Software costs
- Backup and recovery processing
- Upgrades to hardware and software
- Server utilization
- Security
- Network

In 1995, the Gartner Group published a report (K-LAN-308: Dave Cappucio; February 27, 1995) in which they looked at the relative costs of multiple distributed servers versus a smaller number of larger, centralized servers.

They compared the systems management and operational costs of six distributed servers to two larger centralized servers. Their data is shown in Table 6 on page 22.

Table 6. Distributed and Central Server Costs (Gartner Group, 1995)

Support Discipline	Quality of Distributed Service	Distributed	Central	Savings
Storage Management	Medium	$6,928	$5,774	$1,154
Backup	Low	$15,604	$10,403	$5,201
Administration	Medium	$4,763	$4,330	$433
Hardware Activity	Medium	$2,886	$1,443	$1,443
Trouble Shooting	Low	$5,774	$2,887	$2,887
Performance Tuning	Low	$5,052	$2,526	$2,526
Monitoring	Low	$4,763	$4,330	$433
Total		**$45,770**	**$31,693**	**$14,077**

For this case, the centralized environment is 31% less expensive than the distributed server environment.

Another case study can be found in *Selecting a Server - The Value of OS/390*, SG24-4812 where a centralized UNIX server solution was 19% less expensive than a distributed one.

2.4.2.2 Consolidating Multiple Servers onto a Smaller Number of Servers

Once servers have been centralized, the next step is to *merge* them. This entails moving workloads from many small servers to fewer large servers. There are several important issues to consider prior to consolidating workloads. Customers must consider the ability to mix workloads, different levels of service availability, security and reliability requirements, and peak workload implications.

Assuming that these conflicts can be addressed, there are significant benefits from merging workloads. The data in Table 7 was taken from the Gartner Group report referenced in 2.4.2.1 *Centralizing Servers into One Location*, which also looked at the benefits of merging workloads.

The data shown in Table 7 is based on a three-year cost of ownership.

Table 7 (Page 1 of 2). Comparison of Server Costs (Gartner Group, February 1995)

Large Server Cost	$62,000
Small Server Cost	$14,000
Management Tools per Server	$3,900
Tape drive cost per Server	$2,200
Backup Software per Server	$1,200
Storage Cost per Mbyte	$1.7
Storage Average Annual Growth	30%
Technical Salary	$58,000

Table 7 (Page 2 of 2). Comparison of Server Costs (Gartner Group, February 1995)	
Administrative Salary	$38,000
Overhead Burden	40%
Hours per Year	1,950
Technical Hourly Rate	$42
Administrative Hourly Rate	$27

The hourly wage rate is calculated by multiplying the salary by the overhead burden and dividing by the number of work hours per year (for example, for technical support ($58,000 * 1.4) / 1,950 = $41.64). The large server is more than three times the price of the small server because Gartner priced a server with additional capacity and built-in reliability and redundancy features.

The Gartner Group report compared the hardware and support costs of six small servers and two large servers. They assumed that the servers were co-located, therefore, there is no additional cost to centralize these servers. They also assumed that the workload can be merged. Gartner suggested that two servers can be supported by one individual. A potential saving that Gartner did not include is software license costs.

The comparison shown in Table 8 is for the three-year cost of ownership for each case.

Table 8 (Page 1 of 2). Gartner Group Comparison of Six Small Servers to Two Large Servers

	Six Small Servers	**Two Large Servers**	**Savings**	**% Total Savings**
Servers				
Backup	$20,40	$15,600	$4,800	1.3%
Network Card	$9,000	$4,000	$5,000	1.5%
NOS	$89,666	$68,885	$20,781	5.6%
Upgrades	$30,000	$20,000	$10,000	2.7%
Depreciation	$84,000	$124,000	-$40,000	-10.7%
Disk Subsystem	$122,000	$108,528	$13,472	3.6%
Management Utilities	$46,800	$15,600	$31,200	8.3%
Vendor Maintenance	$25,200	$37,200	-$12,000	-3.2%
Server Total	**$427,160**	**$393,813**	**$33,347**	**8.9%**
Support				
Storage Management	$103,936	$43,307	$60,629	16.2%
Backup Servers	$187,264	$62,421	$124,843	33.2%
Administration	$77,952	$28,373	$49,579	13.2%
Hardware Activity	$25,984	$17,323	$8,661	2.5%
Troubleshooting	$51,968	$17,323	$34,645	9.2%

Table 8 (Page 2 of 2). Gartner Group Comparison of Six Small Servers to Two Large Servers				
	Six Small Servers	**Two Large Servers**	**Savings**	**% Total Savings**
Performance tuning	$45,472	$30,315	$15,157	4.0%
Monitoring	$77,472	$30,315	$47,157	12.6%
Support Total	**$570,528**	**$229,377**	**$341,151**	**91.1%**
Total Cost	**$997,688**	**$623,190**	**$374,498**	

Thus the additional benefit of reducing the number of servers, following centralizing, is:

- Server costs - 8%
- Support costs - 40%
- Total costs - 38%

Note: This case study used data and costings from 1995. Since that time, server hardware costs have fallen, while there is on ongoing increase in support costs. Overall the conclusions of the case study will remain valid, since the support cost savings far out weighed those from the server hardware and software.

2.4.2.3 Consolidating Multiple Systems onto One System

There are two possible IBM solutions to further consolidating distributed workloads onto a single centralized system: the IBM RS/6000 SP, and the IBM S/390 Parallel Enterprise Server.

In the remainder of this section, we will look at both of these cases.

Justifying the IBM RS/6000 SP versus Centralized Servers: The following justification is based on IBM experience with multiple customers.

Several customers indicate that it takes one systems administrator to adequately support four centralized database servers. For this example, we assume that systems administrator labor costs are about $10,000 per month, including benefits and overhead. If the administrator can support four database servers, that equates to $2,500 per month per server.

One IBM customer consolidated 24 UNIX servers, each running an independent Sybase application. This customer moved their workloads to a 24-node RS/6000 SP. The robust systems management capabilities of the SP allowed the customer to reduce its systems administrator staff by two people. In addition, by exploiting the SP's ability to isolate systems administration commands by specific users, this customer was able to deploy lower cost operational specialists to do some of the more mundane systems administration tasks. Assume that an operational specialist costs $8,000 per month, including benefits and overhead. Therefore, the total staff to support the SP was two systems administrators and two operational specialists. This customer example will be used in the analysis that follows.

This example is based on the Gartner Group costings and RS/6000 list prices in 1995. The following assumptions were used for equipment costs in the financial analysis:

- RS/6000 Model 590 server with 128MB of memory and a 2.2GB F/W disk
- Additional costs for software

- Depreciation over three years using the straight line method
- $600 for monthly maintenance for each server

Two cases were considered:

1. 24-node SP versus 24 RS/6000 Model 590 UNIX servers
2. 36-node SP versus 36 RS/6000 Model 590 UNIX servers

When all hardware (RS/6000 Model 590 and RS/6000 SP) and software (AIX and PSSP) are compared, the configurations differed in cost by less than 5%.

The SP server consolidation customer referenced in this example increased number of nodes from 24 to 36. The integrated systems management tools packaged with the SP allowed them to increase the number of nodes while maintaining the same support staff. This resulted in the support costs for the SP remaining the same in each case. In addition, it is important to note that customers only pay for the AIX operating system once for the SP versus buying one for each server. These two examples of the SP economies of scale result in a lower hardware and software cost and more importantly significantly lower support costs.

The analysis provided in Table 9 shows the total cost of ownership for the four environments:

Table 9. Comparison of 24/36 RS/6000 Model 590s to 24- and 36-Node SPs

Monthly costs	**Twenty-four 590s**	**Twenty-four node SP**	**Thirty-six 590s**	**Thirty-six node SP**
Hardware/SW Depreciation	$44,400	$42,687	$66,600	$64,662
Maintenance	$14,400	$10,350	$21,600	$15,560
Support Labor	$60,000	$36,000	$90,000	$36,000
Total	**$118,800**	**$89,037**	**$178,200**	**$116,312**

For the 24-node case, the monthly cost difference is $29,763 in favor of the SP configuration (25.1% lower cost), while for the 36-node case, the business value is even more compelling ($61,888 per month, or 34.7% lower costs).

This analysis shows the power of the integrated systems management software packaged with the SP. As server consolidation configurations grow larger, the justification of the SP becomes more significant. This is clearly displayed by comparing the total cost of computing savings of the 24 CPU example to the 36 CPU example.

Following are other examples of potential savings that were not included in the analysis:

- The ability to reduce the required number of nodes by utilizing tools such as the loadleveler to drive up the effective utilization of the processors.
- The savings of not having to buy systems management tools for the SP.
- Reduced networking costs due to the internal network packaged with an SP when using the high performance switch.

Justifying the IBM S/390 versus Centralized Servers: The IBM S/390 has a logical partitioning function called the Processor Resource/Partitioning Manager (PR/SM) that enables the customer to allocate CP, memory and I/O among up to 15 independent operating system images (for example, OS/390).

A description of PR/SM capabilities versus UNIX vendor products can found in the ITSO Redbook *Selecting a Server - The Value of OS/390*, SG24-4812.

PR/SM enables the total processor resources to be applied to the running of the separate operating system images.

Thus, in the case of server consolidation, the workloads from up to 15 separate servers could be migrated to OS/390 UNIX System Services with no change to the mode of operating; communication between servers would still done using the current methods (TCP/IP and so on).

The operating benefits ascribed to the IBM RS/6000 SP as a single machine for server consolidation also apply to the S/390 with multiple logical partitions (LPARs).

If more than 15 servers need to be consolidated to a single machine, then some amount of workload merging will be required. For example, multiple instances of the same application could be run as a single copy of the application against a single, merged, copy of the database.

The following section argues that such merging to a smaller number of operating images brings even further benefit to the consolidation process.

2.4.2.4 Consolidating Servers onto One Operating System Image

Merging multiple servers (single or multiple applications) offers the following additional benefits over those obtainable by running these on a single machine (multiple nodes of an RS/6000 SP or LPARs of a S/390):

- Workload balancing of all applications set by business policy in accordance with business priorities
- A single copy of data to maintain (single, or multiple databases)
- Easier, authorized, access to production mission-critical databases and applications
- Reduced communication between systems
- Opportunity to consolidate both the application server and database server in a three-tier client/server application with consequent improvements in reliability and performance

In the event that the total power required for the merged systems exceeds that of the largest single S/390 processor (currently the 9672-RY5), then the S/390 systems can be clustered and still appear as a single system to the user.

Also, while a single S/390 processor offers outstanding reliability, and is considered to be the industry leader in this area, customers may also want to have a clustered configuration in order to improve service availability.

Both these requirements are met by the IBM Parallel Sysplex architecture (see 3.5, “Parallel Sysplex” on page 65). Up to 32 S/390 processors can be clustered in a single Parallel Sysplex, which continues to provide a single image of an application, no matter how many systems it is using. A Parallel Sysplex can also provide near-continuous availability of service to the end user as it offers the capability to

provide uninterrupted application access during both unscheduled hardware and software failure, and scheduled periods of maintenance or configuration change when most UNIX systems would have to be taken out of service.

A single image Parallel Sysplex also requires a lower operational and technical support headcount than a similar number of unconnected servers.

Some idea of the additional benefits can be obtained from the case study in *Selecting a Server - The Value of OS/390*, SG24-4812. This case study was intended to demonstrate the lower three-year cost of ownership of the latest S/390 offerings with OS/390 versus both distributed and centralized UNIX servers. As such, it was not intended as a Server Consolidation exercise.

However, we can use some of the numbers as a very conservative illustration of the benefit of both a single S/390 and a Parallel Sysplex of S/390s as the target of merging the workloads from a number of centralized UNIX servers.

In Table 10, three cases are shown:

- A base configuration with four applications, each with approximately 100GB of user databases (and servers of comparable processor capacity)
- The same four applications with double the database size (200GB per application)
- Quadruple the database size (400GB per application)

The data from this study are shown in the table:

Table 10. Single Image OS/390 Server Compared to Centralized UNIX Servers

	Base	**Double capacity**	**Quadruple capacity**
Centralized UNIX	$10.536M	$18.006M	$33.360M
S/390 Single Image	$9.704	$15.744	$26.123M
S/390 Configuration	SMP Server	Parallel Sysplex	Parallel Sysplex
Benefit of S/390	**7.9%**	**12.6%**	**21.7%**

As can be seen, the additional benefit of a single image OS/390 server over any other form of server consolidation increases as the total size of the consolidation increases.

The benefit of IBM's Parallel Sysplex clustering is clearly shown in these figures.

2.5 Is My Application a Good Candidate for Server Consolidation?

Decisions made during the initial phase of any project can have a fundamental effect on quality, customer satisfaction, costs and the overall success of the subsequence engagements.

When an enterprise is considering an application(s) for server consolidation, there needs to be a very early assessment of the costs, effort, and benefits of performing the consolidation.

Also, it may not be immediately evident *which* type of server consolidation will be best for the enterprise:

- Physical movement of the distributed servers to a centralized location
- Movement of the individual servers to nodes of a single centralized machine such as an IBM RS/6000 SP, or to multiple logical partitions (LPARs) of a S/390 Enterprise Server
- Complete integration of the server applications and data into a single S/390 Enterprise Server

There are several methods that can be used to go forward. Many of these methods could be right, depending on the circumstances. The objective is to ensure that the correct method is selected, proper expectations are set and that a viable high-level plan is created. These objectives will be used to define a path with the appropriate steps, projects and proposals.

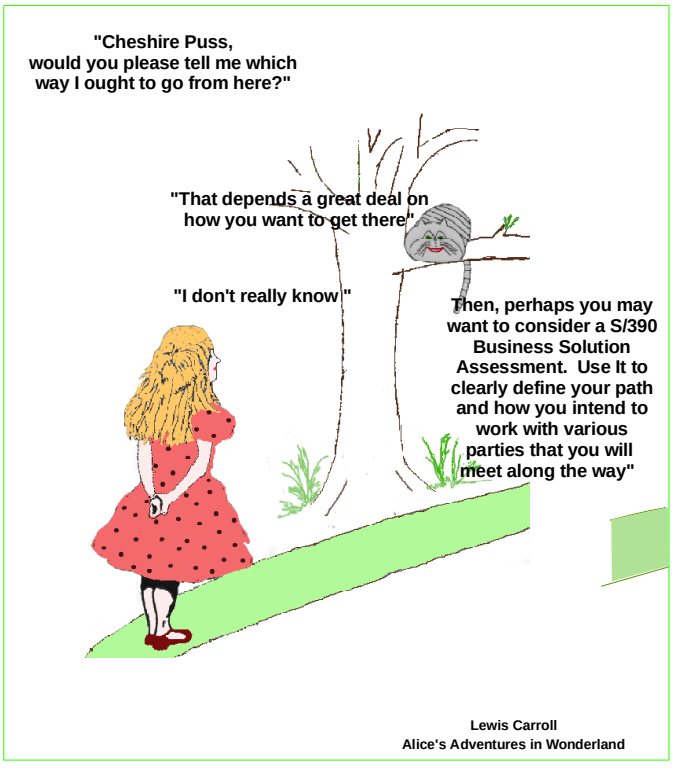


Figure 7. Making Informed Decisions

As an analogy, when Alice in Figure 7 met the Cheshire Cat in the woods, there were many obstacles in her path and she did not know which way to go. She was in an area that she did not totally understand and there were many difficult decisions to make. Alice was afraid to make a decision on how to continue her journey.

The Cheshire Cat's answer made her began considering how she was going to reach her destination. The Cheshire Cat's advice offered Alice a *method* to begin her journey. Now that her position is clear and the obstacles removed, she can consider the right path to get to where she would like to be and decide the appropriate parties to take along.

Similarly, IBM's S/390 Division has developed a methodology known as a Business Solution Assessment that can be used to assess the viability, effort and benefits of many types of application migration to S/390. Additional detail can be found in Appendix A, “Consolidation Candidate Selection” on page 177, or by consulting the IBM Client Representative for the enterprise.

Following the Business Solution Assessment, and once the position is understood, customers can determine the path to take, the likely results and the best projects to suit the work ahead. IBM and third parties will be able to understand the components required to present a unified solution to the customer, and to better define and agree to a workable project plan.

Chapter 3. Why Consolidate to S/390

Distributed UNIX client/server environments are a key component in corporate computing, and UNIX is projected by industry observers to continue to be a premier development platform well into the next decade. IBM has created an organization (S/390 Partners in Development), tools, and an environment that over 1,000 software application vendors are using to port UNIX applications to S/390 enterprise servers.

OS/390 UNIX System Services (an integral component of OS/390) has passed X/Open's test suite of 26,000 tests and has been branded as UNIX 95.

It is a straightforward process to move UNIX source code to OS/390 and to recompile the application, establishing a new production environment on the S/390 which is transparent to the end user and which preserves the UNIX look and feel.

From the user's point of view, it is the best possible environment to run their applications providing:

- High performance
- A high bandwidth production environment
- Open access to alternate platforms, including Windows NT and Novell Netware clients

As the true cost of distributed applications is becoming better understood, IT organizations are increasingly being called upon to manage all of the servers in an enterprise, or are actively seeking out this role. An IT organization could see order-of-magnitude savings in labor costs through improved systems management, lower maintenance, and fewer failures. At the same time, the need to constantly add new servers for more capacity can be alleviated as S/390 can scale to handle more work than any other commercial system.

Over the past few years, the S/390 platform has not only been transformed into a remarkable new system, but also has become a driving force in transforming enterprise-wide computing strategies. The new, smaller and more open S/390 offers many IT organizations a new solution: a hybrid environment that combines the best features of decentralization with the strengths of centralization.

The strengths which set S/390 systems apart from any other operating system - scalability, reliability, availability, capacity, security and manageability, have been retained and enhanced. The price performance for S/390 server hardware and operating systems has been dramatically improved so that S/390 can participate in the client/server world and compete equally with alternative UNIX and NT platforms. The need for virtually continuous computing is being met with IBM's Parallel Sysplex architecture (see 3.5, “Parallel Sysplex” on page 65).

The ability of a S/390 Server to run hundreds of integrated applications, sharing common data, without planned or unplanned system outages, provides an enterprise with the ability to service its customer's needs in a highly responsive and cost-competitive manner.

3.1 Benefits of S/390 over UNIX

Note: Much of the information in this section is extracted from *Selecting a Server - The Value of OS/390*, SG24-4812. That book contains additional information on the differences between UNIX and OS/390. Only information relevant to server consolidation is discussed in this section.

The following are areas where the S/390 with OS/390 can offer a superior environment for consolidation of multiple servers into a centralized single system:

- Improved availability of service to the end user
- High transaction rate systems requiring both superior transaction performance and scalability to handle large volumes
- Security and integrity to prevent unauthorized access to enterprise data, while enabling the use of operational data in the emerging area of electronic commerce
- Improved manageability:
  - Operational
  - Storage management
- Ability to access operational data in existing S/390 systems, and to be able to integrate applications which ran in disparate systems in the distributed UNIX world
- Workload management to enable the efficient concurrent running of formerly separate, distributed applications
- Applications requiring high I/O bandwidth
- Backup/recovery and disaster recovery

These benefit areas of the S/390 Enterprise Server and OS/390 are summarized in Table 11. The functions in this table are discussed in 3.3, “S/390 Strengths as a Centralized Server” on page 35.

Table 11 (Page 1 of 2). OS/390 and S/390 Functions which Provide Superiority over UNIX

Benefit from Server Consolidation	**IBM S/390 or OS/390 Function**
Availability	Hardware fault tolerance
	Processor hardware error recovery
	Software error recovery
	Non-disruptive installation of hardware and microcode changes
	Ability to non-disruptively change hardware configuration
	Parallel Sysplex
Performance	High performance hardware instruction set (CISC)
	OS/390 Workload Manager
	Parallel Sysplex workload balancing
I/O bandwidth	S/390 channel architecture & design
	OS/390 and DFSMS/MVS I/O balancing, recovery & management
Scalability	Efficient SMP processor design
	OS/390 resource management
	Parallel Sysplex
Security	Hardware storage protection

Table 11 (Page 2 of 2). OS/390 and S/390 Functions which Provide Superiority over UNIX	
Benefit from Server Consolidation	**IBM S/390 or OS/390 Function**
	Integrated Cryptographic Facility (ICF)
	Resource Access and Control Facility (RACF)
Integrity	Hardware design
	OS/390 integrity guarantee
	Resource Access and Control Facility (RACF)
Operability	Hardware Management Console
	OS/390 automated operations
	AOC
	NetView
	Tivoli TME 10
Disk storage management	DFSMS for disk management
	Tape Mount Management
	DFSMS Removable Media Manager
	IBM Virtual Tape Server
Workload management	OS/390 Workload Manager
	SmartBatch for OS/390
Backup/Recovery	I/O bandwidth
	ADSM
	OS/390 and subsystem utilities
	Automated operations
	IBM SnapShot for the RAMAC Virtual Array
Disaster recovery	ABARS
	Multiple backup copies with DFHSM
	IBM SnapShot for the RAMAC Virtual Array

3.2 OS/390 and LAN Data Integration

OS/390 is a logical safe haven for data that is currently scattered in Local Area Networks (LANs) throughout the enterprise. When all corporate data resides on the same robust system, the more reliable and efficient server can deliver an immediate payback to the end users.

- Data sharing throughout the enterprise is vastly simplified.
- Consolidation allows high levels of security and data integrity that are nearly impossible to achieve in a distributed environment.

In many client/server infrastructures, centralizing LAN data can bring dramatic improvements in data transfer speed. Existing S/390 connectivity features have helped many of our customers to cut typical transfer times for large files from hours to minutes.

Porting LAN data to S/390 is a completely transparent process. No changes are required in LAN server or client workstation programming code, and there are no visible changes at the level of the client's GUI.

Access to S/390 resources gives LAN users a powerful new suite of facilities including:

- High-volume storage
- Automated backup and data management
- Access to high-speed printers

OS/390 LAN Services is an integrated component of OS/390 and consists of two major functional components (which were formerly separate, independent offerings):

1. LAN Resource Extension and Services (LANRES) provides disk serving, print serving, data distribution, and central administration for Novell NetWare LANs.

   LANRES services include:

   - Disk Serving allows workstation hard disks to be stored as single files on OS/390. To the end user, there is no difference between files stored on an OS/390 disk and files stored on hard drives on the NetWare server.

     Using the OS/390 environment for disk serving can increase the disk storage capacity available to NetWare file servers while providing the reliability of S/390 disks. LANRES has the ability to add volume segments to the NetWare file server without shutting down the file server. When the S/390 hardware data compression facility is used, more information can be stored on the centralized disks, thus resulting in additional cost savings.

   - Print Serving provides for both LAN-to-host and host-to-LAN printing.

     One advantage of using LANRES for LAN-to-host print services is that LAN clients can use high-speed, high-volume host printers for faster turnaround and do it in a physically secure environment.

   - Data Distribution allows authorized OS/390 users to manipulate files and directories controlled by NetWare. The volumes containing these files may be on the NetWare server or on a LANRES host disk. This function implements a central data distribution capability and allows customers the ability to create their own applications for moving data between the OS/390 system and the LAN.

   - LAN Administration allows authorized OS/390 users to perform LAN administration tasks. Connectivity options include direct channel attachment, SNA LU6.2, and TCP/IP. All NetWare clients are supported.

2. LAN Server

   The LAN server component of OS/390 LAN Services provides OS/2 workstation clients with transparent access to OS/390 resources. It also provides an efficient NFS file serving capability for the OS/390 environment.

   Briefly, the LAN server is designed to:

   - Provide a OS/2 workstation compatible file system on S/390
   - Provide high performance and transparent access to S/390 resources
   - Support transparent sharing of data between clients of LAN file services
   - Reduce the LAN administrator's workload by leveraging S/390 resources and services

- Allow the large storage capacity of S/390 to be used to relieve the capacity constraints of workstation-based servers
- Support both LAN and coax-attached workstations
- Support ESCON and IBM PS/2 Micro Channel to Mainframe Connection channel attachments between S/390 and the OS/2 LAN Server

In a recent edition (mid-1997), Computerworld recently conducted a customer satisfaction survey for distributed servers of all types, which suggests the kind of benefits an IT organization might expect from centralization. Computerworld editor Paul Gillin wrote: "Mainframe users are happier with their vendors and products than users of distributed systems, across the board. In fact, the lowest customer satisfaction score in mainframes is higher than the highest score among PC or UNIX server users."

According to Computerworld, "Recent surveys show most large sites are determined to recentralize data to some degree." Increasingly, the trend toward bringing distributed data back to the data center is driven not only by the obvious savings in administration and systems management, but because recentralized storage facilitates data mining.

With business intelligence as a top initiative in many corporations, the fact that OS/390 and DB2 in a Parallel Sysplex cluster can handle the largest databases as a single image becomes a critical advantage. It eases the job of building and maintaining a coherent data warehouse. It runs data mining and decision support workloads efficiently and at relatively low cost, giving users continuous access. Therefore the three corners of the business intelligence triangle, warehousing, data mining and decision support, are a natural outgrowth of a program that begins with LAN data integration.

3.3 S/390 Strengths as a Centralized Server

In this section, it will be assumed that there is a business case for server consolidation. Thus here we will take it for granted that a server consolidation is justified and therefore the need for such requirements as higher service availability, the ability to handle mixed workloads, the ability to provide highly scalable performance, and so on, will be understood by the enterprise.

The features and attributes of S/390 running OS/390 which contribute to its claim to be the platform of choice for server consolidations to either a single machine, or single system, can be summarized under the following areas. Each of these areas are expanded in the remaining sub-sections of this chapter. Note that some features and functions contribute to multiple areas of strength and have their own separate sub-sections.

Areas of S/390 Strengths

- Availability of service
  - Server hardware fault tolerance and error recovery
  - Spare server engines
  - OS/390 software error recovery
  - Non-disruptive installation of hardware and microcode changes and upgrades
  - Clustering with IBM Parallel Sysplex (see 3.5, "Parallel Sysplex" on page 65)

- Security of data and applications
  - Hardware storage protection
  - Cryptography
  - Resource Access Control Facility (RACF) (see 3.3.6, “RACF” on page 49)
- Integrity of data and applications
  - OS/390 integrity guarantee
  - Resource Access Control Facility (RACF) (see 3.3.6, “RACF” on page 49)
- Server Performance and scalability
  - Efficient SMP design
  - OS/390 resource management (see 3.3.7, “OS/390 Resource Management” on page 50)
  - OS/390 Workload Manager
  - OS/390 batch workload management
  - Parallel Sysplex Workload Balancing (see 3.5, “Parallel Sysplex” on page 65)
- I/O performance
  - S/390 channel architecture and design
  - OS/390 data-in-memory
  - OS/390 I/O balancing, file allocation, recovery and management (see 3.4, “System-Managed Storage” on page 51)
- Operability
  - Hardware management console
  - Automated operations software (AOC and so on)
  - Automated network operations software (Netview)
  - Automated systems management operations software (Tivoli TME 10)
  - Operating systems management of disks and tapes (see 3.4, “System-Managed Storage” on page 51)

3.3.1 Availability of Service

The philosophy of the UNIX and OS/390 operating systems with regard to providing service availability are highly divergent.

OS/390, and the underlying S/390 server, have evolved in response to customer insistence that everything possible should be done by the hardware and software to prevent an interruption to service.

This means that a failure which affects one application *must not* bring down the entire system, as there will be other users of that system.

UNIX systems, on the other hand, have focused on the ability to provide fast restart after a failure. This technique, while quite appropriate for small, single application systems, is totally inappropriate when there are an increased number of both users and applications running on a large, centralized server.

Availability is built into S/390 hardware and OS/390 software design, as illustrated in Figure 8 on page 37.

Most modern UNIX systems, can only claim a service class of 99.9% (three 9s), while even fault-tolerant systems generally only rate as a class of 99.99% (four 9s) in this categorization.

Service Class Definitions

Service Class (No of 9s)	Availability	Unscheduled Outage per Year
5	99.999%	5 minutes
4	99.99%	53 minutes
3	99.9%	9 hours
2	99%	88 hours
1	90%	876 hours

Service Class Definitions

Platform	Service Class
Parallel Sysplex	5
	4
S/390 and OS/390 with XRF	3+
S/390 and OS/390	3-

Figure 8. Service Availability Classes Definition and Platforms

Some of the key contributors to the outstanding service availability of an OS/390 system running on S/390 are outlined in 3.3.1.1, “Server Hardware fault tolerance and error recovery” together with a summary of the benefits to server consolidation.

3.3.1.1 Server Hardware fault tolerance and error recovery

The basis for all high availability systems is of course the hardware fault tolerance of the underlying server.

Over the past three decades, S/390 has developed a large number of techniques and functions that make it one of the most reliable server architectures on the market. Some of these are:

- Hardware error handling.

  S/390 server hardware provides Error Coding and Correction circuitry that can trap transient hardware problems as they occur, diagnose the fault and retry the failing operation transparently to the application.

- Hardware error retry/recovery/logout/call home.

  In the event of both transient and permanent errors, the error will be logged out by hardware, and software retry/recovery can be attempted.

  For a permanent error, the hardware will automatically notify IBM via telephone of the situation, also sending diagnostic data, and a service engineer can be automatically dispatched to fix the problem.

  Even when a component “hard fails,” if there is redundant hardware the system/application operation may be moved (by either hardware, software, or a combination of both) to another, still-functioning component (for example, another processor engine, channel, or area of memory). Thus while the error is detected and reported, the application continues to run.

- Memory ECC and spare chips.

  An example of error correction and redundancy can be found in IBM S/390 memory. The ECC codes are capable of detecting and correcting all single and double bit errors, and most multiple bits errors are corrected.

  Should a faulty area of memory develop, there are spare memory chips which can be used to bypass the faulty area.

Such techniques are generally superior to typical UNIX machines. An exceptional UNIX processor in this regard is the IBM RS/6000 S70 which possesses similar error correction and detection capabilities, error logouts and redundant components (such as power supplies) to the S/390.

Benefit for consolidated systems: There will be more systems/applications after consolidation, therefore we cannot afford an outage due to hardware malfunction, which might have been acceptable to a single UNIX server.

3.3.1.2 Spare Server Engines

The S/390 9672 Enterprise Server is an SMP processor with from one to ten engines.

Even when the customer only purchases a uniprocessor (1-way SMP), the server always contains a number of additional processor units (PUs).

There is always one additional PU, known as the System Assist Processor (SAP), which is used to offload I/O handling from the main processor and which also controls services and error handling for the server.

There is also at least one spare engine (apart from the 10-way server, where there is insufficient space on the processor board). This can be used for:

- An additional SAP for configurations with abnormally high I/O handling requirements (for example, airline reservation systems).
- A spare CP which can be brought into service to replace a failing engine (in most cases this is done without operator assistance). This is known as *CP Sparing*.
- To maintain uninterrupted service to an application even if the engine on which it is running fails (application preservation).
- To provide a Integrated Coupling Facility (see 3.5, “Parallel Sysplex” on page 65) instead of a separate independent Coupling Facility (CF), thus offering lower costs to a clustered configuration.

Benefit for consolidated systems: Spare engines maintain performance, availability, and total SMP capacity.

Typical UNIX systems will fail or run at lower capacity as they do not have this capability. This could have a big impact for consolidated server systems.

3.3.1.3 OS/390 Software Error Recovery

The ability to survive transient or permanent hardware failures is only part of the requirement for high service availability to applications.

Systems software is, by its very nature, highly complex. Often there may be millions of lines of code, and unexpected combinations of events can result in even the best-tested code failing.

OS/390's predecessor MVS/370 recognized this as long ago as 1973, when the foundation for today's software error recovery mechanisms was created.

Every major module in OS/390 provides its own Function Recovery Routine (FRR) which is automatically invoked in the event of a failure in the module. The FRR then logs out the error to the same log as is used for hardware errors (where it can be used for problem resolution by IBM software experts), the results of the error are cleaned up, and wherever possible the operation is retried.

The same facility is also used by all major IBM subsystems and database management systems to handle software error conditions. Of course, OS/390 also works in conjunction with the S/390 hardware to support hardware error recovery. The net result is that OS/390 with S/390 can achieve the high service class shown in Figure 8 on page 37.

In contrast, every UNIX vendor offers different variants of the operating system, so that there is no standardized method of recovering from software errors, with a consequent lower service class.

Benefit for consolidated systems: As we have previously discussed a number of times after server consolidation has occurred, there will be a greater number of systems/applications in one environment than there were in the distributed system.

Therefore an outage due to hardware or software malfunction, which might have been acceptable to a single UNIX server, is no longer tolerable.

OS/390 with S/390 provides the most reliable foundation for a single centralized server available today.

3.3.1.4 Non-disruptive Installation and Maintenance

A key contributor to the continuous provision of service is the ability to continue application processing during periods of hardware configuration change, hardware maintenance, and microcode maintenance.

S/390 SMP servers can continue processing during maintenance and repair to:

- A processor engine (on 2-way SMPs and larger)
- I/O channels
- Service processor microcode loads and upgrades
- Power units and cooling fans

Additional I/O devices may be installed and brought into operation without stopping application processing.

When S/390 servers are clustered into a Parallel Sysplex, servers may be added to, or removed from, the cluster without interruption to service.

Benefit for consolidated systems After consolidation, outages for configuration change or maintenance cannot be tolerated due to the larger number of concurrent users who would be affected. S/390 and OS/390 provide an environment where configuration change can be performed without such disruptions.

3.3.2 Security of Data and Applications

Obviously all enterprises value the data which is used to run their organizations, so the security of data and applications is a very important consideration. As mentioned previously, centralizing multiple distributed servers (which can often be in an unsecure office environment) can offer a major improvement to physical security. However secure the physical environment may be, unauthorized access and modification represents a far more serious threat to the continued operation of the enterprise.

Once distributed servers have been consolidated to a single system, the number of applications and quantity of data that must be protected is substantially greater than most UNIX systems can handle. A discussion on restricting access to IBM systems, user applications and data can be found in 3.3.6, “RACF” on page 49.

IBM S/390 server hardware and OS/390 offer protection against inadvertent corruption by users who have passed the RACF authentication process.

The following are key items which offer protection in S/390 and OS/390:

- Hardware storage protection

  OS/390 uses S/390 hardware storage protection to isolate OS/390 components from each other and from all users. No user can deliberately, or inadvertently, access processor memory that they are not authorized to use.

  Thus multiple concurrent applications cannot impact the security/integrity of the other applications.

- Cryptography

  Where very high security is required, such as when the system is opened to electronic commerce via the Internet, an enterprise may wish to further protect application code, data and communications using hardware cryptography.

  IBM 9672 Enterprise Servers now offer this capability as a standard function on all models of the latest G4 range (and as an optional feature on the earlier G3 models). The cryptographic chips are duplexed for redundancy and offer very high rate of encryt/decrypt operations (up to 800 per second).

  In contrast, UNIX systems generally offer only outboard units, which offer lower reliability, higher overhead to invoke, and lower performance

  Many enterprises consider that cryptography is mandatory for E-commerce.

- OS/390 integrity guarantee

  Since 1973 OS/390, and its predecessor MVS, have offered an integrity guarantee. This states that if any way exists for a user to gain control in a supervisor state, or to bypass security checking, then this is considered as an operating system error and will be fixed by IBM.

  No other operating system offers this. Each UNIX vendor maintains their own version of the UNIX operating system. Even if one vendor did offer such a guarantee, this would inhibit portability, which is one of the strengths claimed for UNIX.

Benefit for consolidated systems: Consolidated systems require higher levels of integrity than single servers. E-commerce over the Internet requires protection of information within a server (one user versus another). Only OS/390 on a S/390 server can meet these requirements.

3.3.3 Server Performance and Scalability

Server consolidation demands that the centralized server be capable of handling a far increased workload than any of the distributed servers, and that multiple applications each receive the service that the business requires.

As business volumes increase, the server must be able to provide increased performance and offer growth in a near-linear fashion (near-linear scalability).

If this is not the case, then the server will not be a satisfactory target for server consolidation.

3.3.3.1 Efficient SMP Design

As the name implies, an SMP is a single system that contains multiple, symmetric processors. SMP processors share, and have equal access to, all the component subsystems (from main memory to I/O devices).

SMPs manage large commercial workloads very well, and they can offer very good availability of service. However, even though it has multiple processors, an SMP is still a single physical system that can be disabled by the failure of a subsystem (hardware or software) and that must be taken out of service for upgrades or maintenance. In a global 24x365 economy, these planned outages are increasingly unacceptable.

SMPs offer scalability, but are subject to “the law of diminishing returns.” As SMPs scale, the capacity gained with each new processor diminishes. Beyond a certain point (usually somewhere around 10 to 12 processors, depending on design specifics), adding further capacity can actually degrade performance as the processors compete for shared resources.

A cluster, on the other hand, links individual systems (which themselves may be SMPs) in such a way that they create a single, powerful system that can support numerous users at the same time. Put another way, a cluster is a group of systems that work together as one.

A cluster can offer high availability and excellent scalability. If one system, or node within a cluster fails, the others can continue to run. Because the systems within a cluster do not intimately share resources as the processors in an SMP do, new systems can be added without significantly diminished returns.

S/390 servers running OS/390 not only offer superior commercial performance (transactions/second and batch throughput) but display excellent scaling with server size (SMPs from 1- to 10-way).

In addition, if the capacity required exceeds that of the largest single server, up to 32 S/390 servers can be clustered in a Parallel Sysplex, which also exhibits near-linear scalability as more processors are added to the cluster.

Figure 9 on page 42 shows the scalability of the latest IBM G4 Enterprise Server.

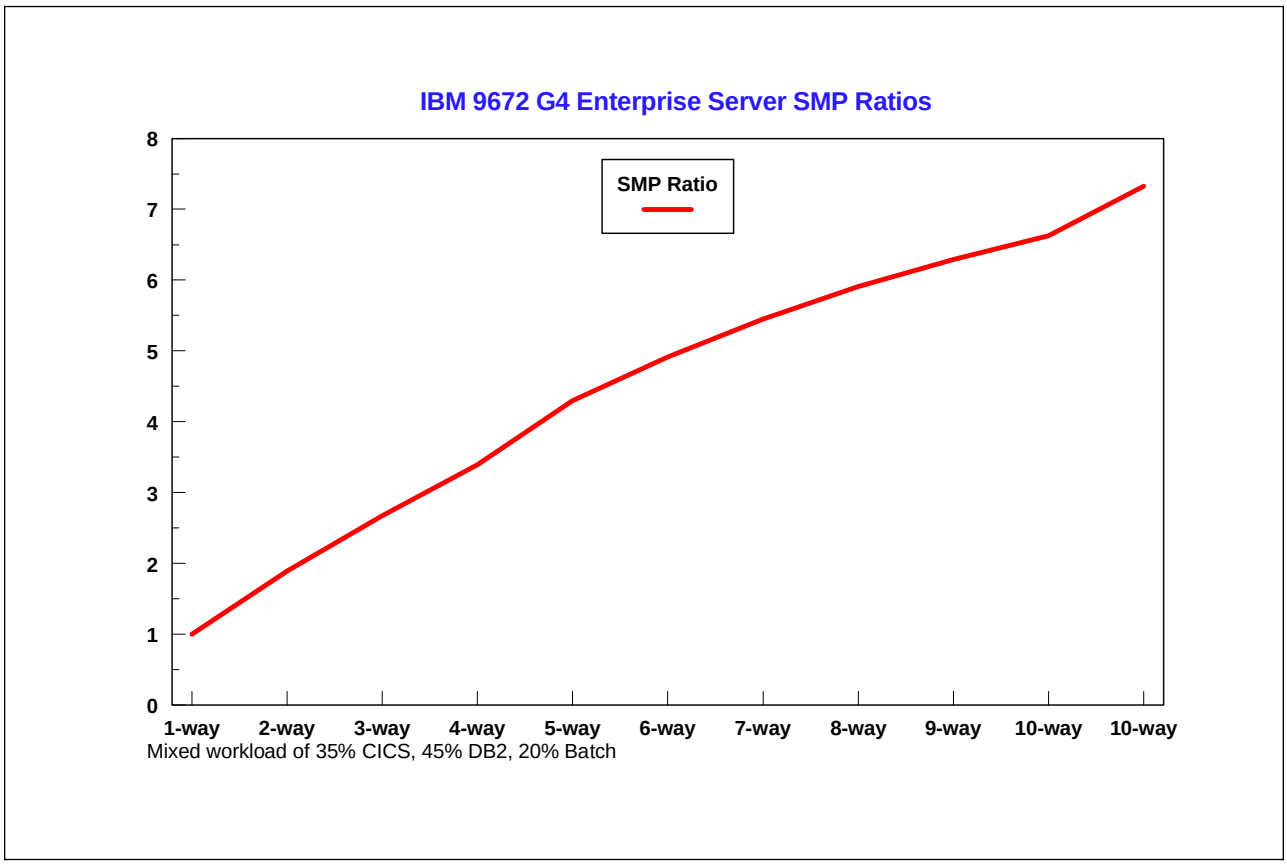


Figure 9. IBM SMP Scalability

Examples of the scalability for both processor- and I/O-intensive relational database workloads (IBM DB2) are shown in Figure 10 on page 43 and Figure 11 on page 43.

Note that the scalability of DB2 for CPU-intensive queries is greater than 94%. (The topic of DB2 performance is extensively discussed in the ITSO Redbook *DB2 for OS/390 Performance Topics*, SG24-2213.)

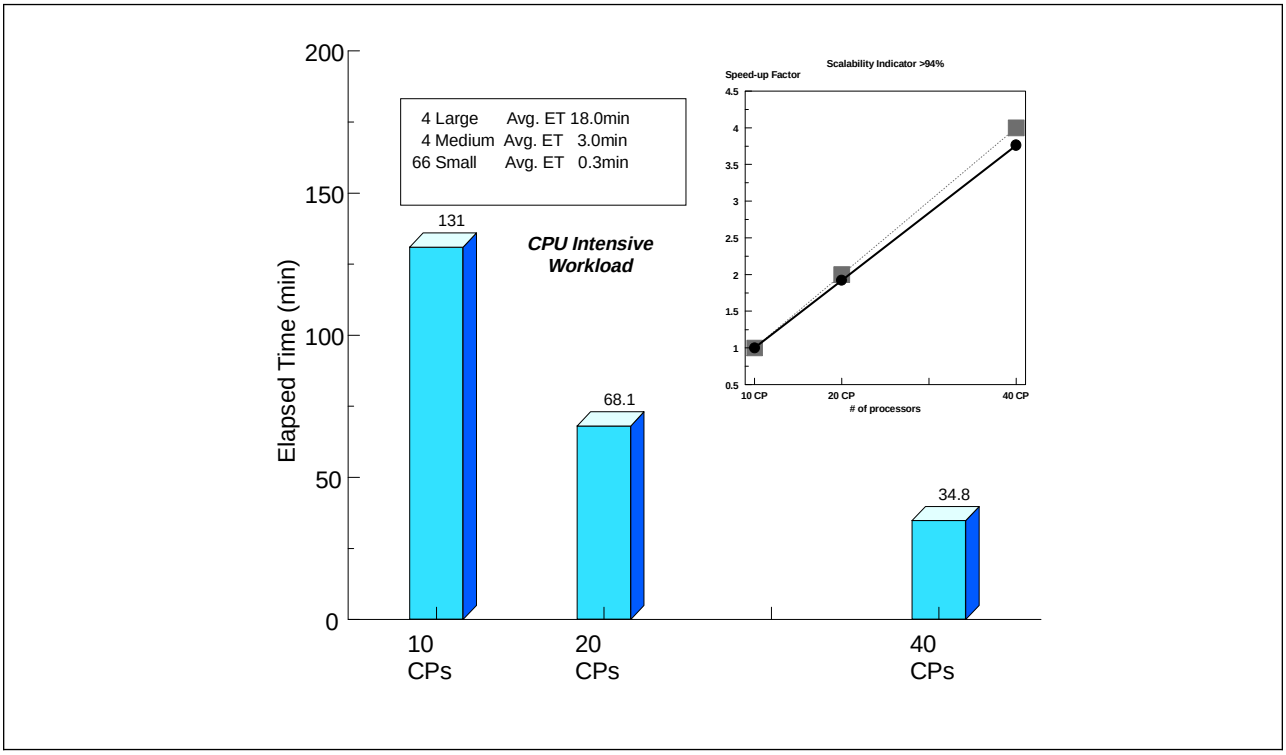


Figure 10. IBM DB2 CPU Scalability

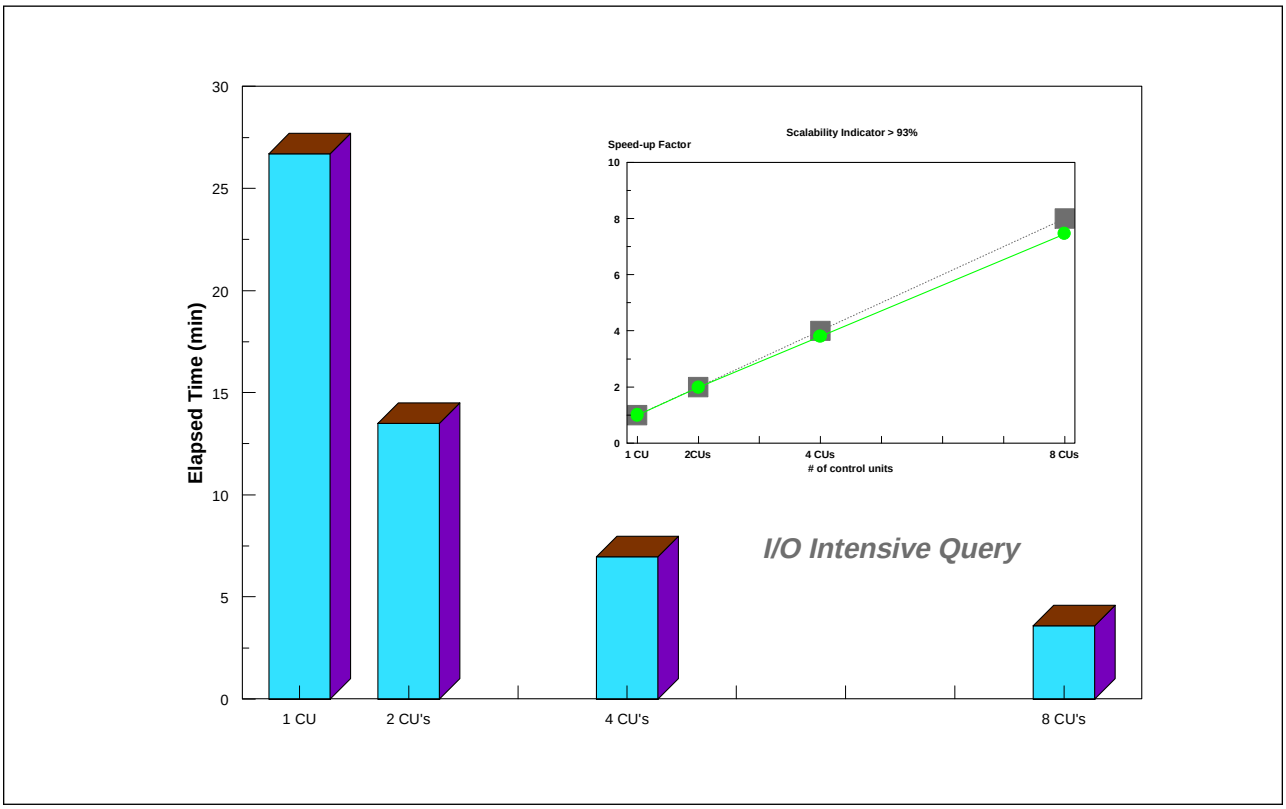


Figure 11. IBM DB2 I/O Scalability

Figure 12 on page 44 shows an example of the scalability of an online CICS/VSAM transaction workload running on Parallel Sysplex cluster.

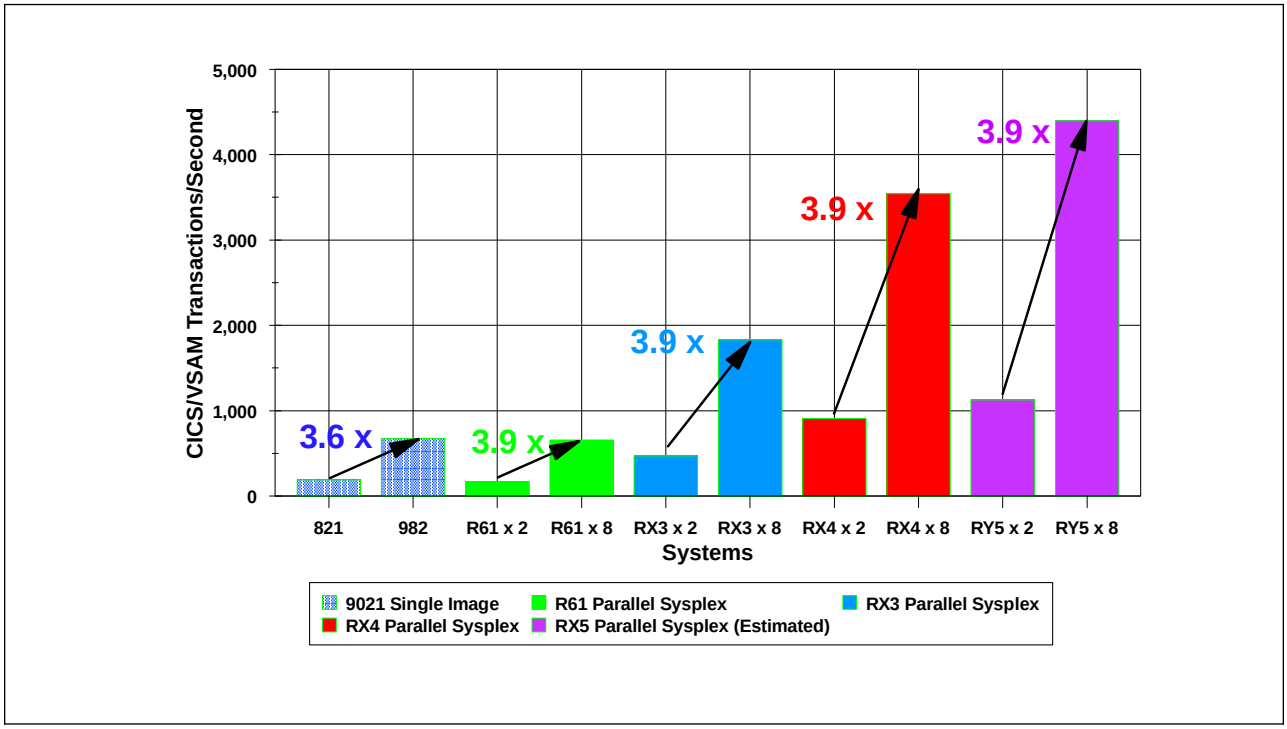


Figure 12. Parallel Sysplex Near-Linear Scalability

Finally, the potential of a Parallel Sysplex cluster to handle peak workloads is shown in Figure 13, which shows nearly 15,000 CICS/VSAM transactions/second on the latest IBM G4 servers:

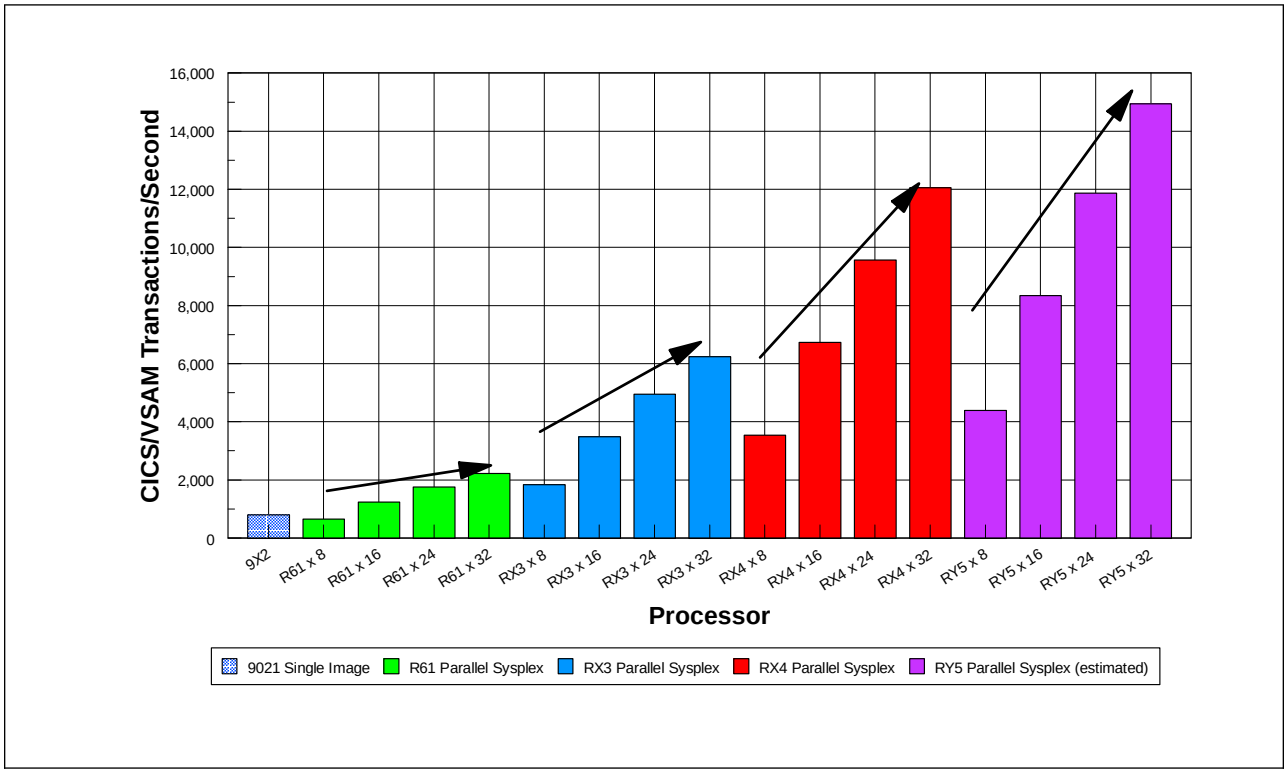


Figure 13. Parallel Sysplex Cluster Performance Capability

3.3.3.2 OS/390 Workload Manager

There will generally be an increased need for batch processing on a consolidated system for the following reasons:

- Larger batches of data
- More data to backup
- Larger databases to reorganize

Much of this will probably be required to be run in parallel with online processing as there is a trend towards longer and longer hours of service in most industries.

Consolidation of multiple applications to a single server also demands a mechanism to provide service to each application in accordance with the business priorities of the enterprise.

With a single system it might be possible to rely on manual tuning, but with a consolidated system the more dynamic environment demands that the system performs this function automatically in response to parameters that describe the installation's requirements.

Thus the ability for the operating system to provide efficient, dynamic workload management is a mandatory requirement.

The companion Redbook *Selecting a Server - The Value of OS/390*, SG24-4812 contains an extensive discussion of the facilities for workload management and scheduling in OS/390 and UNIX. Figure 14 and Figure 15 on page 46 are extracted from that book.

OS/390 offers workload management by policy which is set by the installation according to the business priorities.

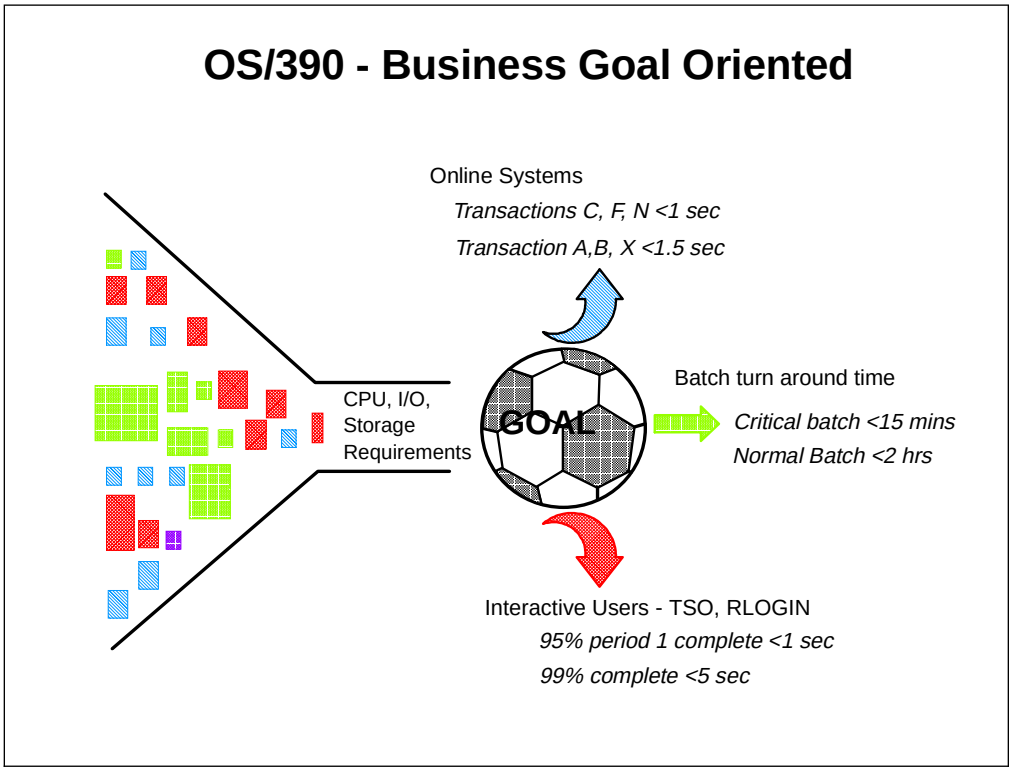


Figure 14. OS/390 Workload Manager

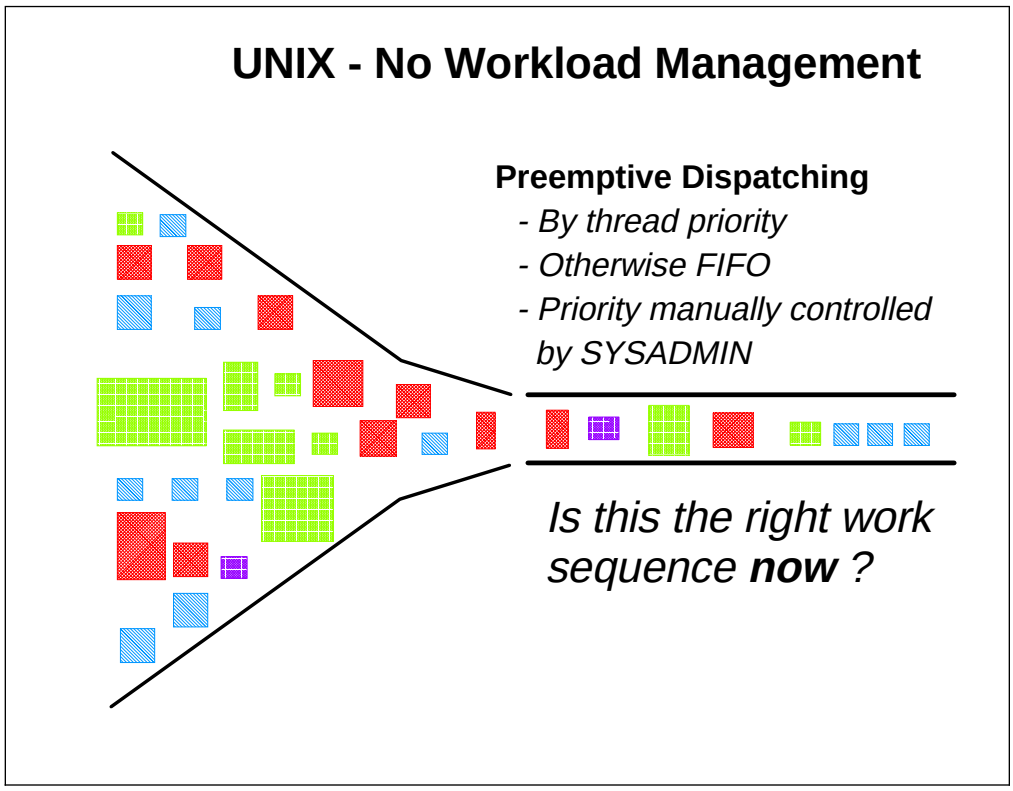


Figure 15. UNIX Dispatcher

Benefit for consolidated systems: The increased capacity required for consolidated systems demands higher levels of resource utilization, greater throughput and near-linear scalability, which is provided by OS/390.

3.3.3.3 OS/390 Batch Workload Management

Workload management while mixed workloads are executing is only one of the requirements for an operating system suitable for large server consolidations.

The increased amount of batch processing expected following a consolidation means that there should also be facilities for managing the scheduling and submission of suites of batch jobs must also exist.

IBM offers a number of tools for this purpose:

- OS/390 itself is well equipped to handle batch job scheduling though its own facilities, for example JES (Job Entry System) scheduling, and WLM dispatching.
- SmartBatch for OS/390 offers a facility that enables the management of suites of batch work across a cluster. In addition, the piping of files between jobs in the same or different servers greatly reduces the elapsed time for a batch suite.
- The facilities of System Managed Storage and SnapShot for the IBM RAMAC Virtual Array (see 3.4.1.4, “Backup/Recovery” on page 61) greatly enhance the performance of the necessary functions of backup and restore.
- Database online online reorganization is provided by IBM database management subsystems (for example, DB2).

3.3.4 I/O Performance

In comparison to S/390, typical UNIX servers are relatively good at at handling processor-intensive workloads, but are relatively poor in I/O bandwidth.

The I/O requirements for a set of distributed UNIX servers may be within the capability of the individual servers, but once server consolidation has occurred, then the accumulated I/O requirements may begin to exceed the capability of most UNIX servers.

One reason for the poor I/O handling of many UNIX designs lies in the fact that I/O traffic must compete with other data traffic within the server on the same bus(es), and I/O processing steals cycles from other processing.

3.3.4.1 S/390 Channel Architecture and Design

S/390 I/O architecture has been based on a design that moves I/O handling outboard from the processor to a channel.

S/390 I/Os are initiated by the processor placing the I/O commands into a hardware-designated area of memory, and issuing a Start Subchannel command. The channel then retrieves the I/O command(s), known as a channel program, and executes the I/O required. This includes movement into/from memory, I/O commands to/from the device, and the handling and retry (if necessary) of error situations.

The result of this design is that I/O processing on S/390 does not compete for processor resources and many concurrent I/Os may be processed at high speeds. The latest IBM G4 Enterprise Server can drive I/O rates of up to 18MB/sec over each of 256 concurrent channels.

The power of this approach (which has been part of S/390 since 1964) is being reflected in the recent initiative among vendors of Intel architecture NT systems in the proposed I2O project (Intelligent I/O).

I2O will bring outboard I/O handing to Intel servers. I/O processing and error handling will be handled outboard. Control will only be passed to the processor when the I/O completes or I/O error recovery is required.

The I2O architecture is expected to first appear in Intel systems sometime in 1998.

Benefit for consolidated systems: Consolidated systems require greater I/O handling than single UNIX servers. Also, while it may be tolerable for an I/O error to bring down an individual distributed server, the same type of error cannot be allowed to impact a consolidated server running multiple applications and wih many more users.

The ability to drive high I/O performance is also critical for high transaction rate online systems, such as those likely to be found on consolidated Web servers running e-commerce.

S/390's greatest strengths are in those areas. In addition, the I/O throughput of the server is matched by the disk and tape I/O subsystems available with S/390 (for example, the RAMAC family of disk arrays, and the Magstar 3590 tape subsystem).

3.3.4.2 OS/390 Data-In-Memory

One very common approach to improving server I/O performance is the motto " the best I/O is no I/O." The idea is that by caching (buffering in S/390 jargon) large numbers of file or database records in server memory, the majority of I/O requests will be satisfied by finding the record in memory.

In such a situation, a "real" I/O will only be required if the record is not found in memory. The record, once read in, will usually replace another record which has not been referenced for the longest time using some form of least recently used (LRU) algorithm.

This technique has been used extensively in UNIX systems by the most popular database management systems. Such systems typically fix large amounts of processor memory, which is then unavailable for other concurrent applications, and mitigates against efficient operating system memory management.

This is not sociable behavior in consolidated systems, where there may be multiple database systems and applications contending for memory.

The popularity of this means of reducing the impact of the poor I/O capability of most UNIX servers, and the resulting requirements for very large memories, is a driving force behind the move to 64-bit addressing in the UNIX environment.

This should be contrasted with S/390 where "data-in-memory" has been an integral part of OS/390 and subsystem design for over 10 years, and the hardware has provided 44-bit addressing since 1985 using a unique design known as Expanded Storage.

This support is transparent to the application and is invoked through the level of buffering specified to the application's database management system. When 64-bit addressing is available for S/390, it is anticipated that its use will also be transparent.

Benefit for consolidated systems: More data-in-memory enables higher application I/O rates and greater system throughput with less processor hardware (memory and channels). This is extremely important when consolidating multiple smaller servers.

3.3.5 Operability

System operations contribute significantly to the costs of running a distributed server system.

In a distributed office environment, this function may be performed by persons on a departmental headcount, rather than being a visible part of the cost of providing the application to the enterprise.

Once server consolidation has taken place, there is the opportunity to not only use fewer operators to perform the same procedures as were previously used in the distributed system, but also greatly improve the efficiency of operations.

Operator efficiency has been a key requirement for S/390-based systems for many years: OS/390, IBM operational and systems management software, and third party software products have evolved to form a solid, cohesive foundation for lowering the cost of operating a large, centralized OS/390 server.

In contrast, since each vendor develops and maintains their own version of UNIX, a similar foundation for automated, efficient operating procedures has not developed to the same extent.

For example, the first customer referenced in Chapter 1, “Introduction” on page 3 employs three people for each new UNIX server they install (one person per server for each of three shifts). This incurs a major expense, which over a three-year period is comparable to the cost of the server!

In contrast, the majority of OS/390 customers operate much large installations (in terms of numbers of applications, users, server processing power, and database size) with a very small number of operators.

A description of the OS/390 and other tools available to the OS/390 installation is beyond the scope of this book. However the following are among the tools available:

- The server Hardware Management Console provides integrated control of both the individual server and a Parallel Sysplex cluster of servers.
- OS/390 can tailor operator console functions so that specific operator actions can be sent to specific operational areas (such as a tape library, for example), or messages from multiple servers can be handled centrally,
- Automated operations software, such as Automated Operator Consoles, (AOC) from IBM can be used to automate mundane processing, or exception handling, without a physical operator being required. This type of support is frequently used to minimize, or eliminate operators for overnight shifts (so-called “lights-out” environments).
- Automated network operations software (such as NetView from IBM) can be used to automatically handle startup, reconfiguration, exception handling and restart after line errors on communications networks.
- Automated systems management operations software (for example, IBM's Tivoli TME 10) reduces the operator involvement in such tasks as problem management, change control, reconfiguration and so on.

3.3.6 RACF

Chapter 8, “Security Considerations” on page 119 discusses some considerations of security when consolidating servers from UNIX to OS/390.

Resource Access and Control Facility (RACF) is IBM's product that addresses the security of the OS/390 environment (note that there are also alternative offerings from third-party vendors that compete with RACF).

The major points of access to OS/390 (and IBM database and network products) invoke the services of an IBM or third-party security product through the use of a standard set of interfaces (The Security Access Facility (SAF)).

RACF provides the following functions:

- User authentication by means of a password. Passwords can be set to expire after installation-specified intervals, and the number of attempts to logon is limited to three, after which the user's access is denied until reinstated by the security administrator.

This is in contrast to most UNIX security systems, which do not enforce frequent password change and which permit unlimited entry attempts. Such systems are the victims of many break-ins by hackers.

- User authority to access system resources (for example, specific files, devices and so on) can be set to various levels of access authority ranging from ownership, through read-only access, to no access at all.
- Users can belong to groups of users with similar needs and authorizations and inherit the authorities of the group. Individual user authorities can be set for specific resources and these override those of the group.
- User groups can be set up in a hierarchy that reflects organizational structures or other business needs.
- RACF can also use industry security standards such as Kerberos, and participate in a Distributed Computing Environment (DCE).

Benefit for consolidated systems: Business today demands that enterprise data be protected from both deliberate and accidental access by unauthorized persons. This requirement is strongly increased when more data and applications are consolidated onto a large, single server which is accessed by a very large number of users. When the needs of electronic commerce are added to this, the security provided by products such as RACF becomes mandatory.

3.3.7 OS/390 Resource Management

Computing based on UNIX systems evolved based on the philosophy that computing resources such as server hardware and devices were inexpensive.

Thus if, for example, a server ran out of capacity for an application, it was considered less costly to simply replace it with another server. An outage to the application would have been required to replace the server.

Similarly when only one application was involved, it was often less expensive to purchase a larger server and run it at low utilization (say 40-50%), than to put the effort into building an operating environment (UNIX, application, database management system, and hardware) which could be run at high utilizations (90-100%).

Low-cost UNIX servers also put computing within the budgets of user departments, and as we have seen, networks of distributed servers evolved.

As previously noted, the hidden costs of distributed servers is leading to an industry-wide trend to server centralization and consolidation. In this centralized environment, where multiple applications and large numbers of users are involved, it again becomes appropriate to look at maximization of the use of resources rather than rapid replacement and obsolescence of equipment.

This is because it may not be possible to support the total workload without driving resource use to higher levels, and the times for outage in order to upgrade a configuration may be limited or non-existent.

Thus in order for a consolidated server to provide:

- High performance
- Scalability
- High service availability

while controlling costs, it is necessary for the system (operating system and associated subsystems) to be able to manage the total system resources (CPU, memory, I/O, devices, and so on) so that high utilizations are possible.

As we discussed earlier in this chapter, OS/390 manages jobs, server processing power and memory to maximize total system throughput in accordance with business policies.

The following section discusses the management of disk and tape storage.

3.4 System-Managed Storage

As a business expands, so do the needs for storage to hold applications and data. Storage costs include more than the price of the hardware, with the highest component being the people needed to perform storage management tasks.

Removable media, such as optical and tape storage, cost less per gigabyte (GB) than online storage, but require additional time and resources to locate, retrieve, and mount.

To allow a business to grow efficiently and profitably, ways have to be found to manage the growth of information systems storage and to use that storage most effectively.

The challenges in the storage and management of data facing an enterprise are illustrated in Figure 16

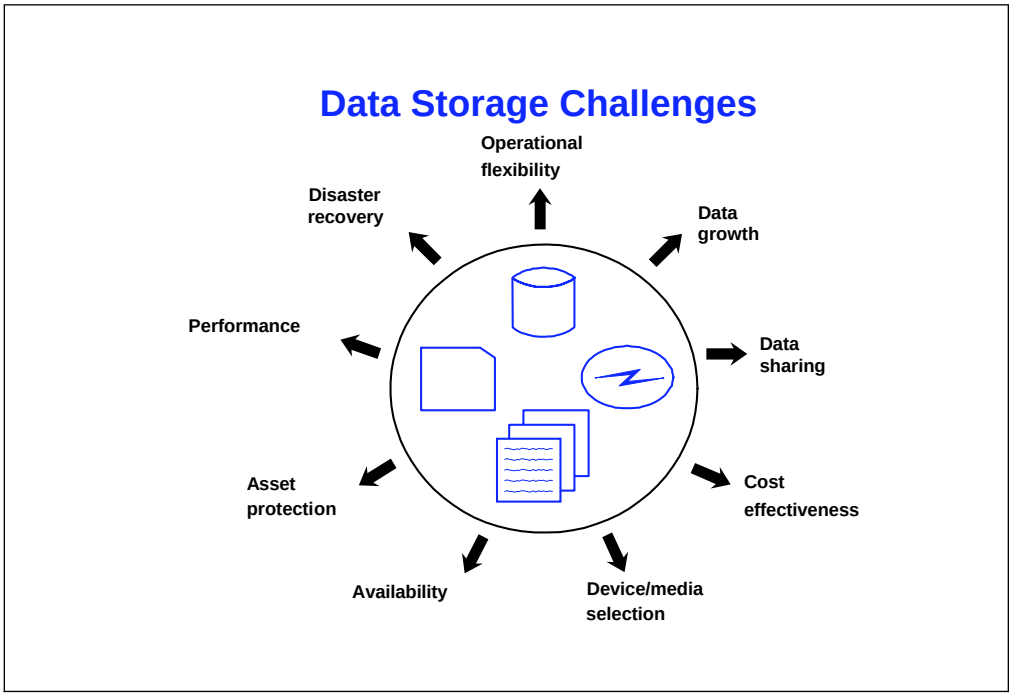


Figure 16. Data Storage Challenges

Storage management must be designed to allow the following:

- Protection of enterprise data assets
- Improved data reliability and recovery
- Consistent storage management across the enterprise

- Productivity improvements for users of that data
- Full utilization of server resources and services

In addition data access must allow application portability, and provide application development flexibility and resource balancing.

All these challenges are faced in each location of a distributed server environment. Additional challenges of data integration and synchronization may also exist if there is also a centralized location involved in data exchange with the distributed sites.

Server consolidation onto a S/390 centralized system offers a means of reducing the cost and effort of the management of enterprise data, while adding to the integrity, security and integrated use of such data.

A co-requisite product to OS/390 is the Data Facility System Managed Storage (DFSMS) product. The totality of the automated storage management function is known as System Managed Storage.

Figure 17 illustrates the manner in which DFSMS, and associated products such as RACF, Data Facility Sort (DFSort) and the AdStar Distributed Storage Manager (ADSM), work together to provide an enterprise-wide storage management framework.

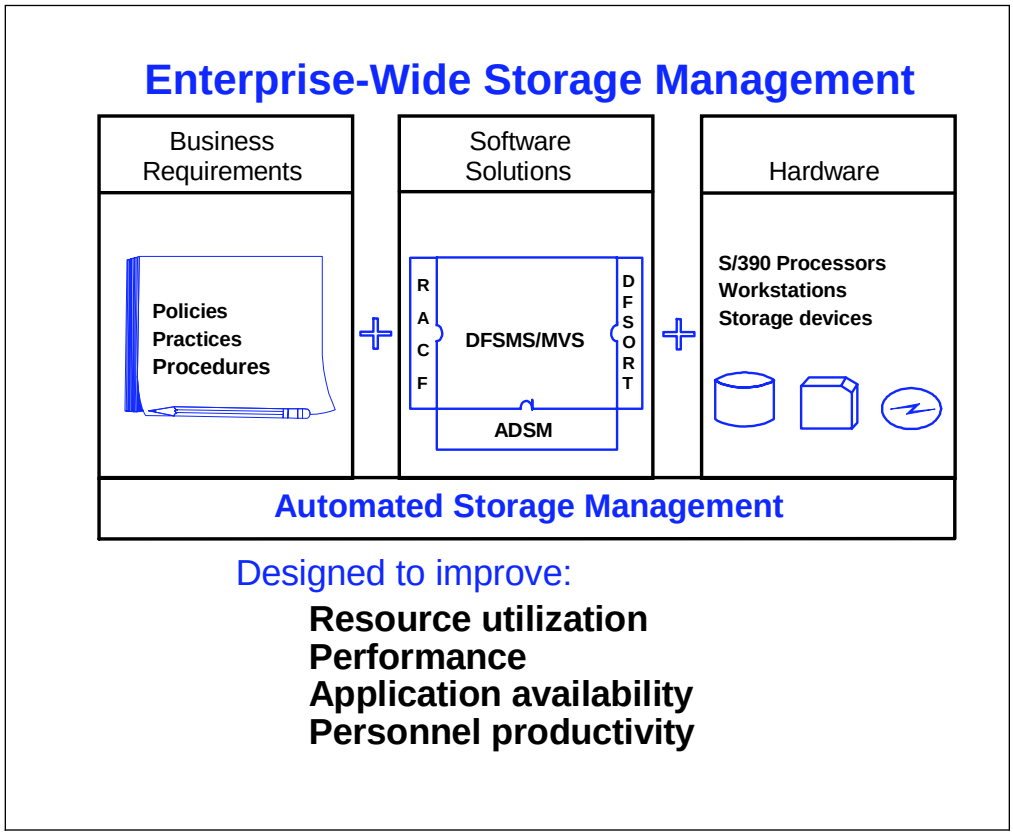


Figure 17. Enterprise-Wide Storage Management

Figure 18 on page 53 illustrates the manner in which policy requirements are applied to the logical and physical management of enterprise data storage.

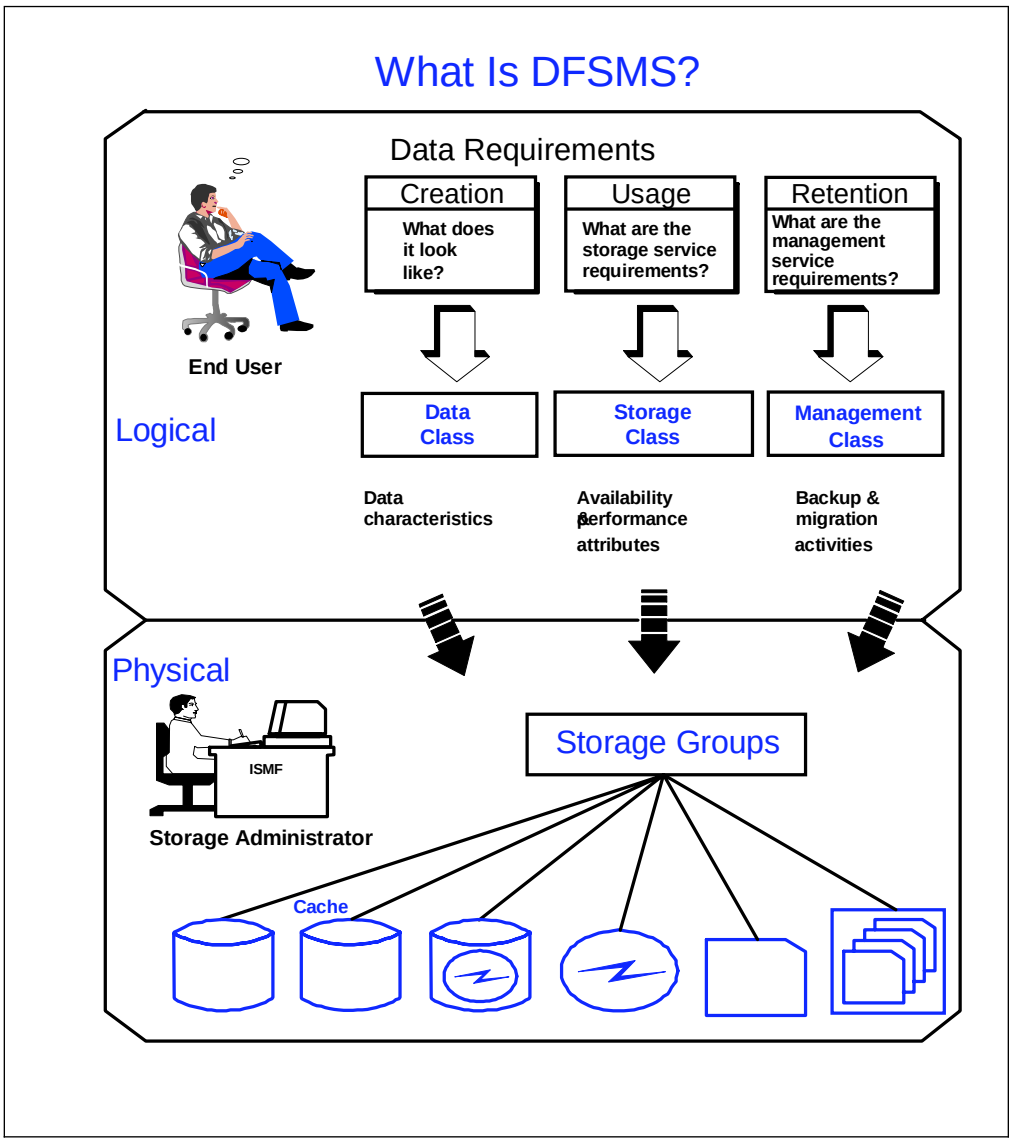


Figure 18. DFSMS Design

System managed storage is provided for both fixed disk storage and for removable media storage such as tape and optical disk. Experience in OS/390 customer environments over many years has demonstrated that this system managed environment brings significantly lower people, hardware and operating costs.

In OS/390, as well as the UNIX file system (known in OS/390 as the Hierarchical File System (HFS)), there are many other data organizations which were developed for the non-UNIX environment.

All the DFSMS facilities and tools are applicable to both the HFS and traditional OS/390 file formats.

Simple centralization of distributed servers does little or nothing, to reduce the cost of storage management since each server has its own data to manage.

Consolidation to a single operational OS/390 system may not only bring economies of scale as individual system disk storage is combined, but also can offer a superior, more cost-effective environment.

The system managed storage functional components that provide value to a consolidated server include:

- Disk storage management

  When each file (including databases) is created, it is associated with a storage management policy (either explicitly, or by default) which enables the business policies of the enterprise to be reflected in the use, access to, and ongoing management of the file (see 3.4.1.1, “Disk Management”).

  System managed storage can perform:
  - Automatic disk space management, including migration of low-use data to other disk, tape or optical media
  - Automatic backup of files at policy-driven intervals
  - Automatic creation of backup copies for storage at an off site location

- Tape management

  System managed storage also can be used to manage tape files, both in production and for backup/restore and disaster recovery. The same controls that apply to the allocation, access and management of disk files also apply to tape files (see 3.4.1.2, “Tape Management” on page 57).

- Distributed data access

  Distributed data access allows all authorized systems and users in a network to exploit the powerful features of system managed storage. The Distributed FileManager (DFM) is used to support remote access to OS/390 data and storage resources from workstations, personal computers, or any other system on a SNA LU 6.2 network.

A more general comparison of OS/390 system managed storage can be found in the Redbook *Selecting a Server - The Value of OS/390*, SG24-4812. The next two subsections expand on the value of system managed storage to UNIX server consolidation to an OS/390 environment.

3.4.1.1 Disk Management

The philosophy of OS/390 system managed storage is that an end user of enterprise data should not be responsible for choosing the optimal device type (disk or tape, high-speed or low-speed disk, and so on), for the control of access by others to the data, the ongoing tuning required for total system performance, or for the day-to-day management of the data (backup/restore, and disaster recovery).

Leaving the protection of critical data to individual users does not necessarily guarantee that data will be consistently backed up or that backup versions will be stored in a safe place. The truth of the matter is that end users often view backing up their data as a nonproductive task that frequently gets put off for another day!

Such tasks and controls should be determined by the enterprise management in accordance to the business needs for access, performance, and management of the enterprise data. OS/390 allows the installation to specify such requirements in the form of policies which are used dynamically to perform the automated system management of data storage.

Some key functions that will be new to users of most distributed servers are:

- Hierarchical Storage Management

OS/390 uses a hierarchy of storage devices in its automatic management of data, relieving end-users from manual storage management tasks.

Disk space usage is improved by keeping only active data on fast-access storage devices. Space on user volumes is automatically freed by deleting eligible files, releasing over-allocated space and moving low-activity data to lower cost-of-storage devices.

- Availability management

  Data is backed up (automatically or by command) to ensure availability in the event of accidental loss of files or physical loss of volumes. A storage administrator can cause the copying of backup and migration tapes, and specify that copies be made in parallel with the originals.

- High speed data movement

  System managed storage utilities allow for the fast movement or copying of data between volumes of both like and unlike device types. This is of value in the consolidation process when data is being imported and merged with data from other servers, and when a server configuration needs to be changed to accommodate changing business requirements or growth.

While there are benefits in consolidating multiple distributed UNIX servers to a single OS/390 environment, this may not always practical, desirable or cost-effective.

Even if consolidation to OS/390 occurs, there may still be applications that should remain distributed for good business reasons.

In such circumstances, it is still highly desirable to extend some of the benefits of system managed storage to the distributed UNIX or NT servers and to PC-based workstations.

The AdStar Distributed Storage Manager (ADSM), an IBM client-server solution for distributed data management, supports a wide variety of IBM and non-IBM platforms for both small and large systems. With ADSM, data on numerous platforms (ADSM clients) can be protected.

ADSM uses a policy management structure similar to that of OS/390's system managed storage. The great flexibility of this policy management structure enables ADSM to decide how to handle each user's data on the basis of defined policies.

ADSM has three major components:

- ADSM Client

  The ADSM client can be a personal computer, a workstation, or a local area network (LAN) file server. The ADSM client holds the critical data that has to be protected.

- ADSM Server

  The client's data is sent to the ADSM server through a communication protocol. The ADSM server stores the data in ADSM storage pools and uses a database and recovery log to track the location of the data in the storage pools. The server can be a mainframe or a workstation.

- ADSM Administrator

The ADSM administrator controls the ADSM server from an administrative client. The administrator also defines the policies that will help protect data in the enterprise. (These policies contain “rules” specifying how data will be treated during a backup or archive operation.)

The design of ADSM is shown in Figure 19.

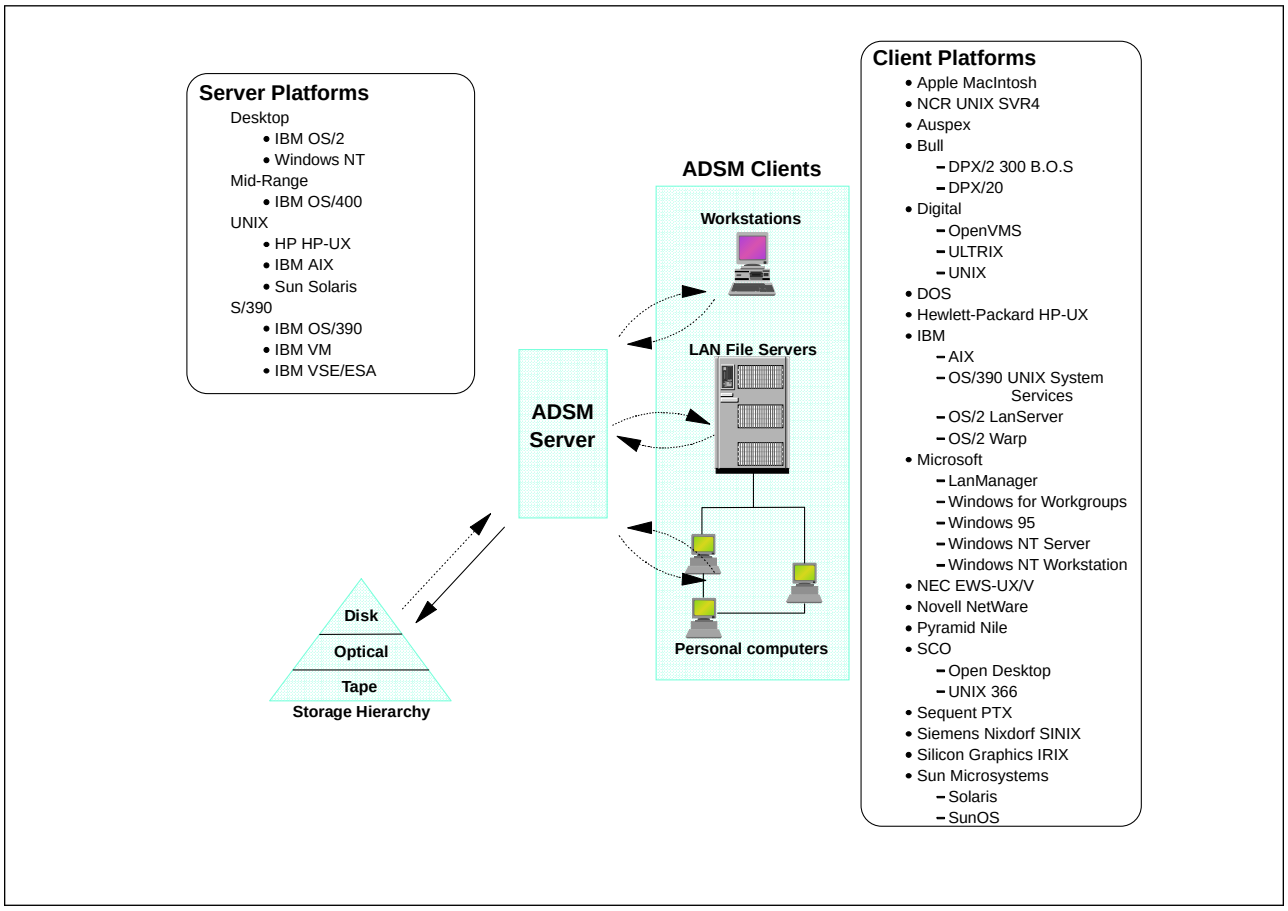


Figure 19. ADSTAR Distributed Storage Manager Design

ADSM automates backup and archival of critical data

Backup saves copies of client files on the ADSM server. Thus, data at the ADSM client is protected in the event of data loss due to a hardware or software failure, accidental deletion, and/or logical corruption. Individual files, directories, or subdirectories can be backed up.

From a client, files can be restored that have been backed up to the ADSM server. During a restore, ADSM copies the backup version from the ADSM server to workstation or LAN server. Restore enables the recovery of a single file or an entire file system.

Archive enables copies of files to be saved for some period of time on the ADSM server. Archive copies are never replaced with more current versions (as is the case with backups).

3.4.1.2 Tape Management

OS/390 system management of tapes offers a number of advantages to the consolidated server. These include:

- Faster backup processes
- Better tape media utilization (fewer tapes mean lower costs)
- Lower operator costs through both fewer tape mounts and through automation of tape operations

Faster backup: A frequent problem of distributed servers is the slow speed of backup caused by the use of low-speed, low-cost tape devices, and by the low I/O bandwidth of many UNIX servers.

Consolidation to a centralized OS/390 server makes the high speed, automated tape drives of this environment available to the consolidated workload.

Improving tape cartridge utilization and tape mount reduction: File, database, and volume backup in many environments (both distributed and centralized) has been associated with individual applications, and backup has not always been done in the most efficient manner.

Studies have shown that frequently only small amounts of data are copied to tape. Often this is the only file on the tape, and large tape inventories have developed of tapes with very little useful data upon them (see Figure 20 on page 58).

The problem is compounded as newer technology tape devices are introduced with ever larger capacity. The unused space represents a large, and avoidable, cost.

The consolidation of multiple, smaller, distributed servers to a single large, centralized server may also cause an increase in the number of tapes used and consequently the number of tape mounts required. System managed storage provides a means for both increasing tape media occupancy, and reducing the number of tape mounts.

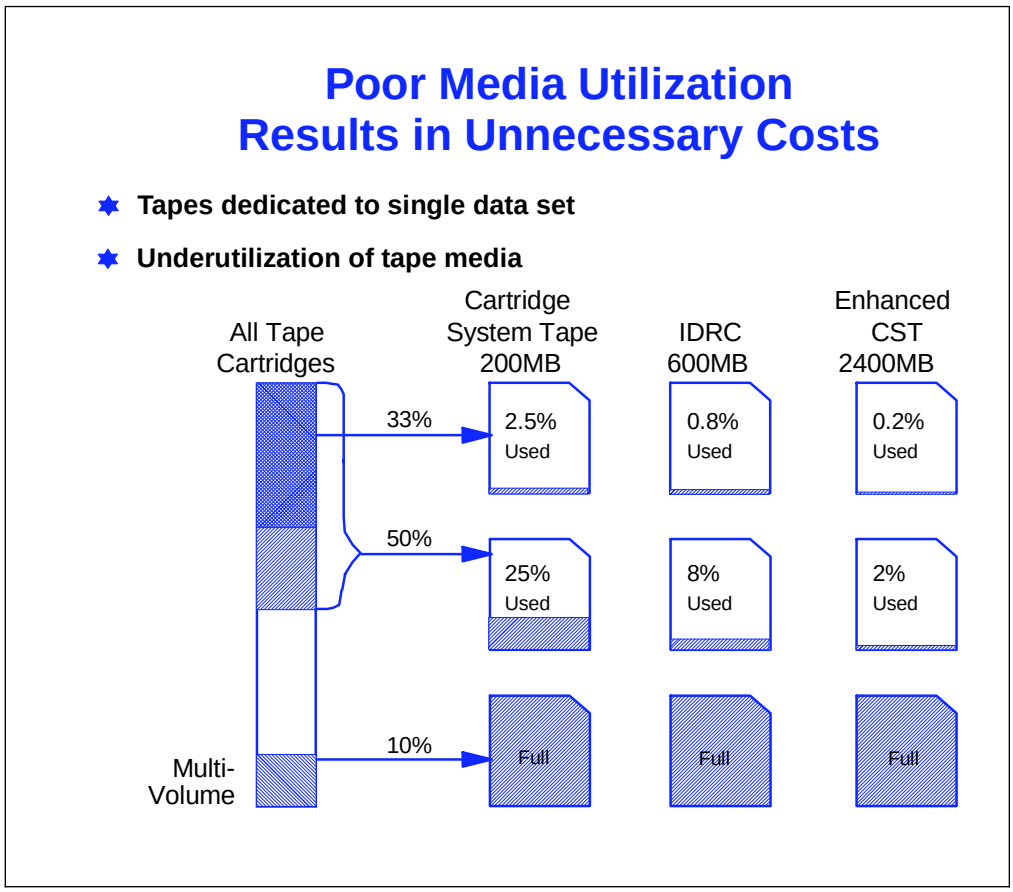


Figure 20. Poor Cartridge Utilization Means Unnecessary Costs

In conjunction with system managed storage, IBM has developed a methodology known as Tape Mount Management (TMM). This, together with continuing analysis and periodic tuning, can lead to significant reductions in the number of tape mounts.

Actual customer experience has shown savings of as much as:

- 65-90% reduction in the number of tape volumes
- 60-90% reduction in the number of tape mounts

The methodology involves intercepting selected tape file allocations through the system managed storage automatic class selection (ACS) process, and redirecting them to a disk buffer.

Once on disk, these files can be migrated to a single tape or small set of tapes, thereby reducing the overhead associated with the mounting of multiple tapes.

Removable Media Management: OS/390 system managed storage provides facilities to manage removable media resources, including tape cartridges and reels. It provides functions for:

- Library management

  Tape libraries or collections of tape media associated with tape drives can be created to balance the work of tape drives and operators.

- Shelf management

Information about removable media is grouped by shelves into a central online inventory, and the use of volumes residing on those shelves is tracked. Optionally, shelf management for storage locations used to vault tapes outside of the tape library is provided.

- Tape management

  The movement and retention of tape volumes throughout their life cycle is managed by system managed storage. Information is maintained about files and tape volumes to validate volume and file information and to help maintain data integrity. Retention of tape files can also be controlled.

3.4.1.3 Virtual Tape Server

The Magstar 3494 Virtual Tape Server (VTS) provides a single integrated hierarchical storage management system that can:

- Lower tape operating costs
- Reduce the number of tape cartridges supporting tape operations
- Reduce the number of tape drives required
- Reduce the tape robotics required for automation
- Reduce the floor space required to store tape volumes
- Increase the level of tape automation

The Magstar 3494 Tape Library automates the retrieval, mounting and demounting, and storage of tape cartridges for host systems.

The Magstar 3494 Virtual Tape Server, an integration of advanced IBM technologies, provides a hierarchical storage management system using Magstar 3590 Tape Drives, IBM fault-tolerant RAID disk storage, a RISC-based controller, and IBM's Magstar 3494 Tape Library to deliver a new class of storage device to the traditional storage products hierarchy.

Through exploitation of the tape and disk resources, this new class of device reduces the floor space required to store tape volumes, reduces tape operating costs, and reduces the cost of tape processing to provide a rapid investment payback.

The logical structure of the VTS is shown in Figure 21 on page 60.

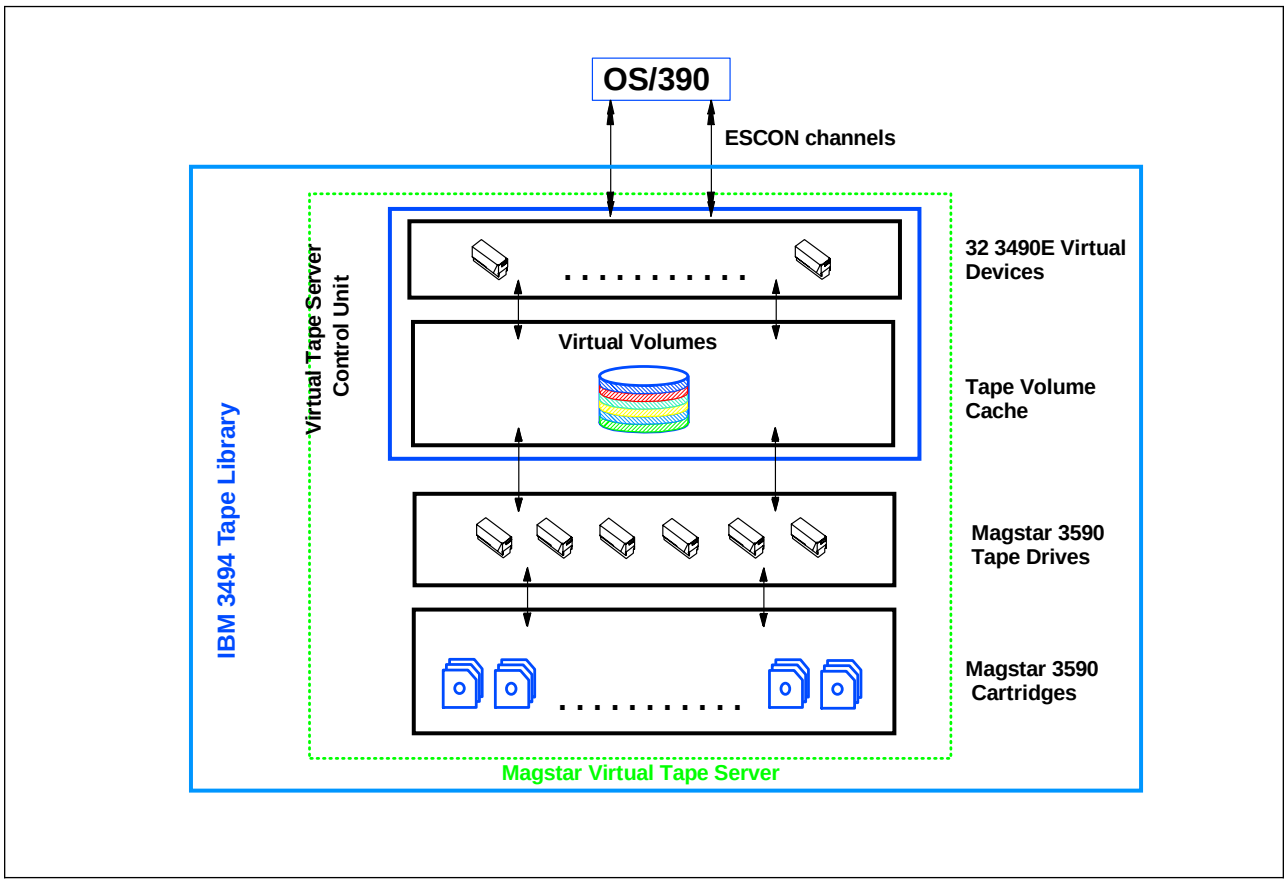


Figure 21. IBM Virtual Tape Server

The VTS offers an automated solution to the operational complexities and wasteful use of resources that may occur when distributed servers are consolidated:

- Inability to fully exploit tape media capacity

  The 3494 Virtual Tape Server integrates advanced tape, robotics, and disk technologies to exploit the 10GB (30GB with 3:1 compaction) capacity of the High Performance 3590 Cartridge Tape and the 9MB/sec data rate of the Magstar 3590 Tape Drive. The VTS presents an image of two fully configured IBM 3490E tape subsystems (up to 32 tape devices) to attached hosts. Data is initially stored as tape volume images (called virtual volumes) on the integrated disk storage. Then, after the tape volume is unloaded by OS/390, the VTS writes the virtual volume to a High Performance 3590 Cartridge Tape (owned by the VTS) with other previously written virtual volumes to form a stacked volume and fill a cartridge with data.

  Previously written tape volumes, called logical volumes, that are stored on a tape cartridge, will be moved back to integrated disk storage when referenced by an attached host processor.

  Demounted logical volumes residing on the integrated disk storage can be remounted for immediate access without actually mounting the tape cartridge containing the logical volume. A virtual volume can store up to 800MB of data. Virtual and logical volumes that contain less than 800MB only consume the actual number of data bytes in the disk storage and 3590 cartridge. Large files can span multiple virtual volumes. A copy of the virtual volume is kept in the

integrated disk storage until it is again accessed by an attached host processor or deleted from the disk storage to make room for other virtual volumes.

Based on a sample of customer data, the 3494 Virtual Tape Server can reduce the number of tape cartridges required by as much as a factor of 59 times through exploitation of the 10GB capacity of the High Performance 3590 Cartridge Tapes.

- Inability to fully exploit tape resources

  The Magstar Virtual Tape Server will move entire tape volumes between the Tape Volume Cache and the Magstar 3590 tape drives at one time. The Virtual Tape Server provides bandwidth management to service multiple, open-tape volumes and exploit the 9MB/sec transfer rate of the Magstar 3590. The result is a reduction in the number of required tape drives. (Note that the operating system still “sees” up to 32 devices.)

- Inability to reduce tape storage and operating costs

  By placing multiple logical 3490E tape volumes onto a single 3590 tape volume, the number of required tape cartridges and tape drives and automated storage capacity is greatly reduced. The floor space occupied by those cartridges, tape drives, and automation systems is correspondingly reduced. The reduction in cartridges, tape drives, and automation also brings a reduction in operations handling to process the workload. This reduction can help lower the total tape operating costs by lowering the purchase price and maintenance of hardware and reducing overall operating costs.

3.4.1.4 Backup/Recovery

The greater amount of mission-critical data that can reside in a single location after server consolidation places an even greater burden of responsibility on operations staff than existed in the distributed world.

In order to complete the backup of all critical data within the minimum time, so that online services are maximized and any batch processing is completed on time, the operating system and server hardware must be able to offer fast and efficient backup/restore processing.

OS/390 running on S/390 servers is renowned for the speed, reliability and efficiency of the backup/restore utilities of both OS/390 and key IBM-supplied database management systems.

These abilities result from the combination of the S/390 channel architecture, which permits multiple, concurrent, high-speed channels to simultaneously transfer data without processor overhead, and the ability of OS/390 to dynamically schedule multiple, concurrent workloads.

IBM SnapShot for the IBM RAMAC Virtual Array (RVA): A major enhancement to the backup and restore process has recently become available for users of the IBM RAMAC Virtual Array (a high performance, RAID-5 plus disk array).

IBM SnapShot for the RVA enables a copy of an entire volume to be created in only a few seconds. As is shown in Figure 22 on page 62, no data movement is involved, only a new set of pointers defining the new volume is created.

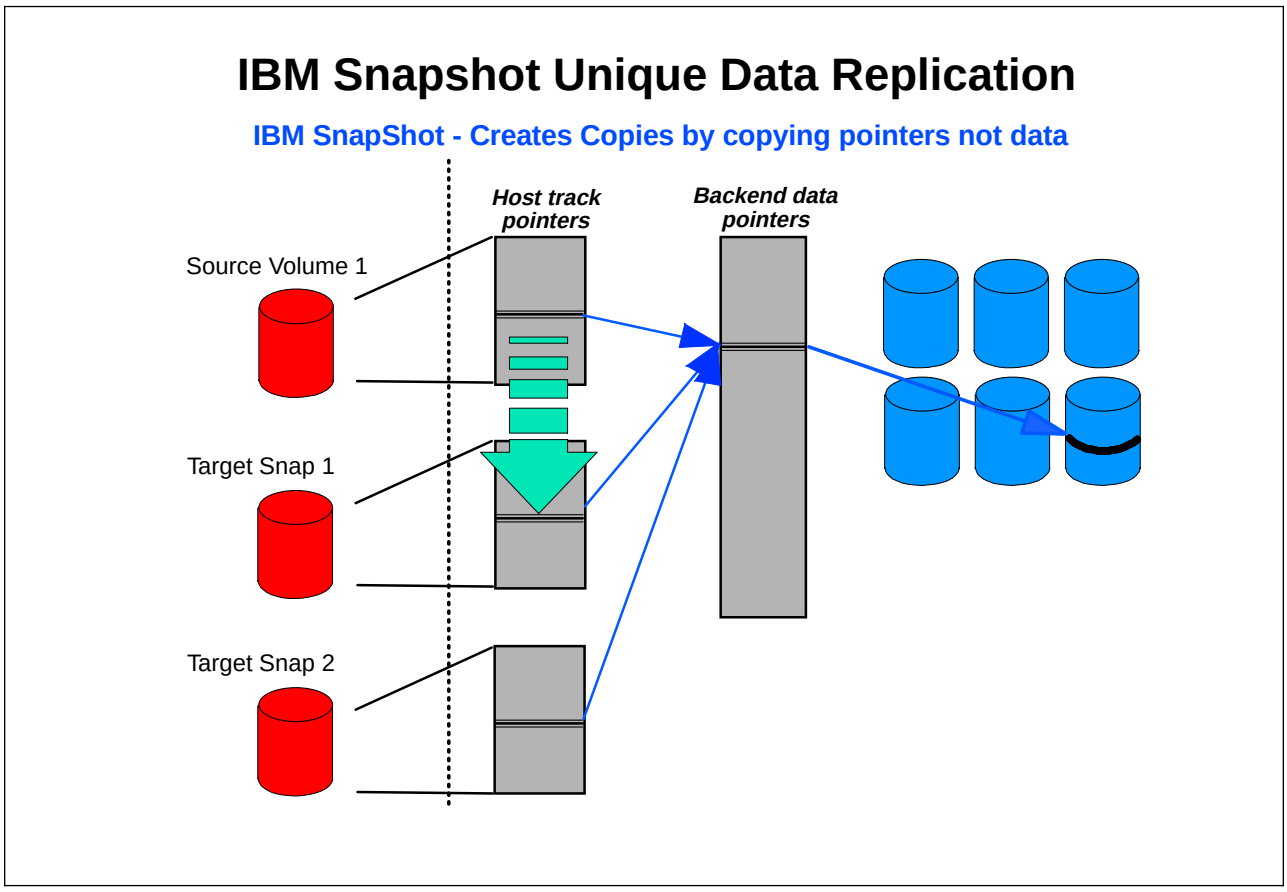


Figure 22. IBM SnapShot for Data Replication

Thus, for example, online processing need only be quiesced for a few seconds to create a backup volume, a volume of test data, a copy of data for batch processing and so on.

As the new volume's data is changed or updated, new data extents are written, leaving the original volume unchanged.

The process is fast and reliable, and quickly repeated if, for example, batch processing fails.

Figure 23 on page 63 and Figure 24 on page 63 illustrate the savings that one customer saw in the backup and batch processes by using SnapShot.

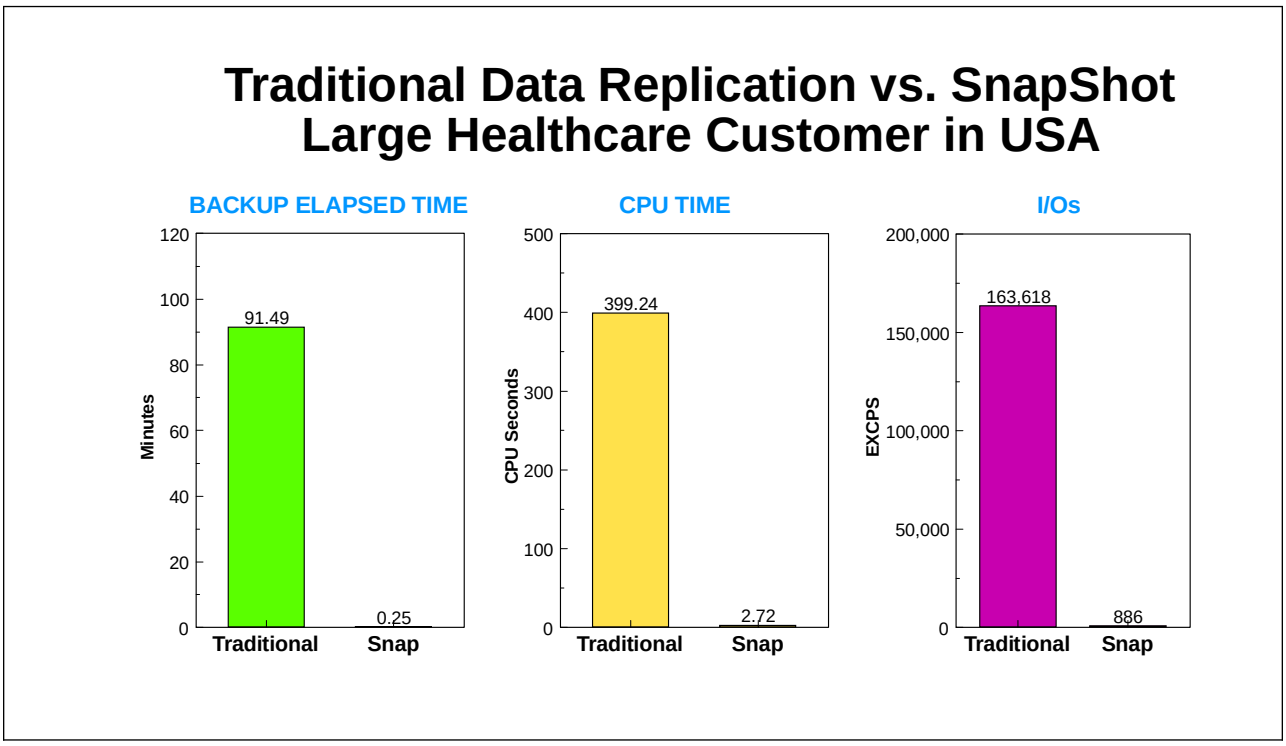


Figure 23. Reduced Backup Times Using IBM SnapShot

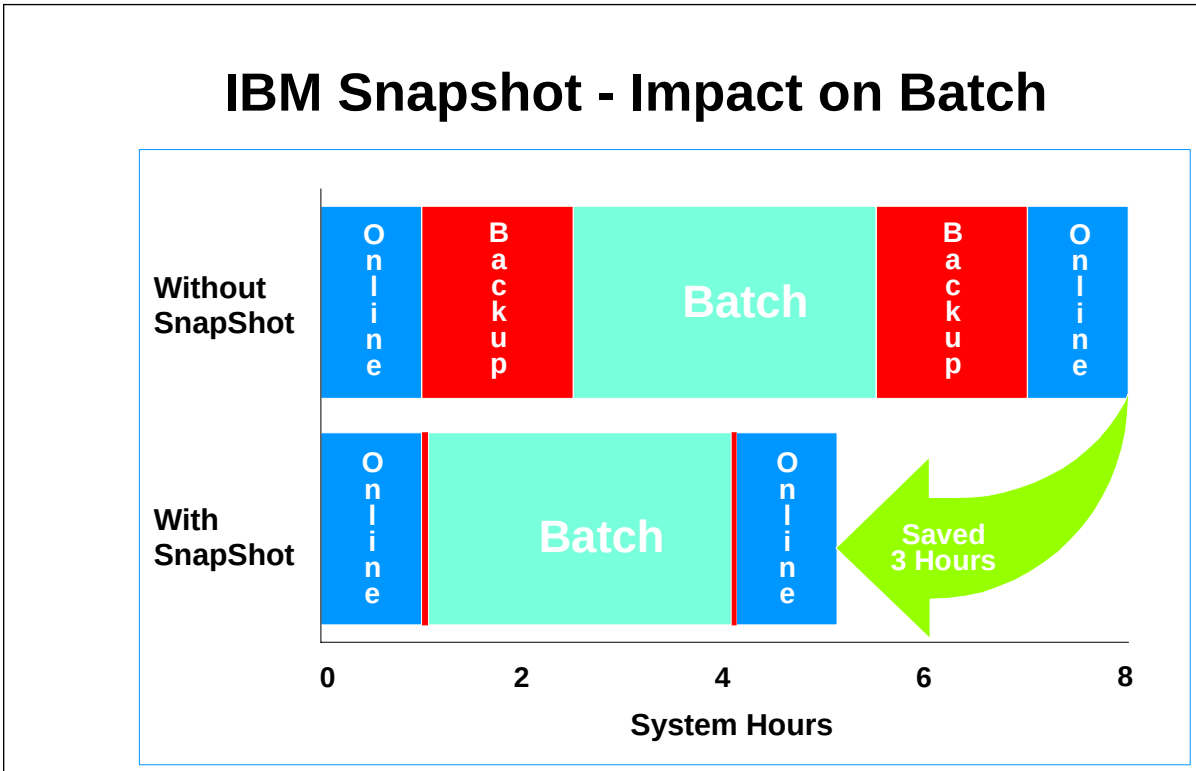


Figure 24. Shorter Batch Processing with IBM SnapShot

3.4.1.5 Disaster Recovery

In a distributed server environment, a natural or man-made disaster will probably not affect all servers equally. Some may be unaffected, while others will be lost completely.

The impact of such a disaster will depend heavily on how good the backup/restore procedures are, how well any disaster recovery procedures have been designed, and how well, if at all, they have been tested.

Implementing and testing disaster recovery procedures for multiple, distributed sites could be a very difficult, if not impossible, task. This is because of the difficulty of knowing and coordinating the state of data in each location, and in coordinating the activities of a large number of persons geographically dispersed, many of whom may not be well-skilled in IT procedures.

Centralizing servers into one location can make it possible to use one set of trained operators with skills obtained over many years of managing central systems.

In this world, it should be easier to develop the required procedures, test data, and actually perform a realistic test of a disaster scenario.

OS/390 and Disaster Recovery: In response to many years of customer experience and requirements, IBM has developed a number of OS/390 system-managed storage and hardware devices that can expedite the development, testing and implementation of successful recovery following a disaster.

These are largely beyond the scope of this book, but include the following:

- A Backup And Restore System (ABARS) is a major component of the products that comprise system-managed storage. This provides a means of collecting all the components of a given workload (programs, job control statements, and files) into a single backup entity that can easily be created, transported and restored at a recovery site.
- Multiple backup copies can be created concurrently for both local and remote recovery.
- Copies of volumes for disaster recovery testing can quickly be created and destroyed using IBM SnapShot for the RAMAC Virtual Array without the expense of acquiring additional disk capacity for this purpose.
- Both disk and tape devices may be attached via fiber optic channels (ESCON) at distances of up to 43km. This means that data in remote, and distinct, data centers may be accessed by applications in both centers. In a disaster recovery situation, this can mean that more up-to-date and accurate data can be resident in the surviving center.
- IBM disk controllers provide for the automatic creation and update of files in a remote location by means of two techniques:

  Peer-to-Peer Remote Copy (PPRC) maintains two copies of a file in a synchronized state during the writing and updating of the file. This means that in the event of a disaster, an exact copy of key files, in their state immediately preceding the failure, is available in the remote site (which may be up to 43 km apart).

  EXtended Remote Copy (XRC) performs a similar function to PPRC, except that the copy process is asynchronous allowing greater distances between

centers. The asynchonous nature of the copy means that the last transactions before a failure may be missing from the remote copy (standard database procedures may be used to resynchronize data using log tapes). The sites may be an unlimited distance apart.

3.5 Parallel Sysplex

As we have described earlier in this book, there are convincing business and technical reasons for an enterprise to consider consolidating multiple distributed servers onto a single-image S/390 system.

However, this consolidation must take into account capacity, performance and service availability characteristics of the consolidated systems.

Centralized UNIX systems frequently address this requirement by clustering multiple servers into a single complex, for example the IBM RS/6000 SP.

Customer requirements have led the information technology industry to redefine what is necessary in a commercial processing system today. Users are looking for systems with dramatic performance and cost improvements that permit them to continue running their existing applications. The rise of the mobile workforce and global connectivity, such as the Internet, have increased the emphasis on high availability as well.

Symmetric Multiprocessors (SMPs) can manage large single commercial workloads well, and they offer good availability. However, even though they have multiple processors, SMPs are still a single physical system that can be disabled by the failure of a subsystem (hardware or software) and that must be taken out of service for upgrades or maintenance. In a global 24x365 economy, these planned outages are increasingly unacceptable.

SMPs offer a degree of scalability, but are subject to “the law of diminishing returns.” As SMPs scale, the capacity gained with each new processor diminishes. Beyond a certain point (usually somewhere around 10 to 12 processors, depending on design specifics), adding further capacity can actually degrade performance as the processors compete for shared resources (see, for example, the latest S/390 9672 processors in Figure 9 on page 42).

A cluster, on the other hand, links individual systems (which themselves may be SMPs) in such a way that they create a single, powerful system that can support numerous users at the same time. A cluster can offer high availability and excellent scalability.

If one system or node, within a cluster fails, the others can continue to run. Because the systems within a cluster do not intimately share resources as the processors in an SMP do, new systems can be added without significantly diminished returns.

Parallel Sysplex is IBM's clustering architecture for the S/390. The design points for Parallel Sysplex were:

- Scalable, granular growth
- Continuous application availability
- Protection of investment for existing applications and systems
- Reduced cost of computing

The challenge in designing clusters is in presenting a single system image - making these separate systems work and appear as one. Typically, with UNIX clusters, the single image is compromised because data is partitioned among the individual nodes, meaning each data partition can be accessed and updated only by the node to which it is connected. The partitioned approach creates significant difficulties in capacity planning, tuning the system for optimum performance and service availability.

From a capacity planning perspective, great care must be taken to divide the data equally across the cluster nodes based on workload activity - otherwise, one node can be overwhelmed with work requests while other nodes sit idle or underutilized. In addition, since each node is responsible for its data, a planned or unplanned outage can impact service availability because a backup node must assume the additional workload, often degrading response times. Data partitioning also may require time-consuming, and costly, database repartitioning and application splitting as work grows or business needs change.

What separates S/390 Parallel Sysplex technology from other cluster implementations is its unique data sharing architecture and OS/390 Workload Manager (WLM).

S/390 parallel data sharing technology allows direct, concurrent read/write access to shared data from all the processing nodes in the configuration - with full data integrity and optimum performance. Given that all the data can be shared, workload is allocated to the nodes, or servers, based on load rather than the location of the data, as in the UNIX clusters described above. This workload distribution is managed by OS/390 Workload Manager (WLM) which, “self-manages” the Parallel Sysplex configuration by dynamically adjusting workload priorities based on customer-defined policies, while optimizing the use of system resources.

The strengths of Parallel Sysplex's clustering approach match the requirements of any business requiring high availability, a clear and simple growth path, investment protection and low total cost of ownership.

The IBM Parallel Sysplex architecture is particularly suited to the requirements of a server consolidation because of its:

- High performance
- High service availability
- The ability to efficiently handle multiple concurrent applications
- The ability to efficiently handle concurrent online and batch workloads (actually a key characteristic of the underlying OS/390 operating system), and the
- Ability to scale to handle very large workloads and high transaction rates

Figure 25 on page 67 illustrates the ability of a Parallel Sysplex to dynamically add capacity to handle workload peaks.

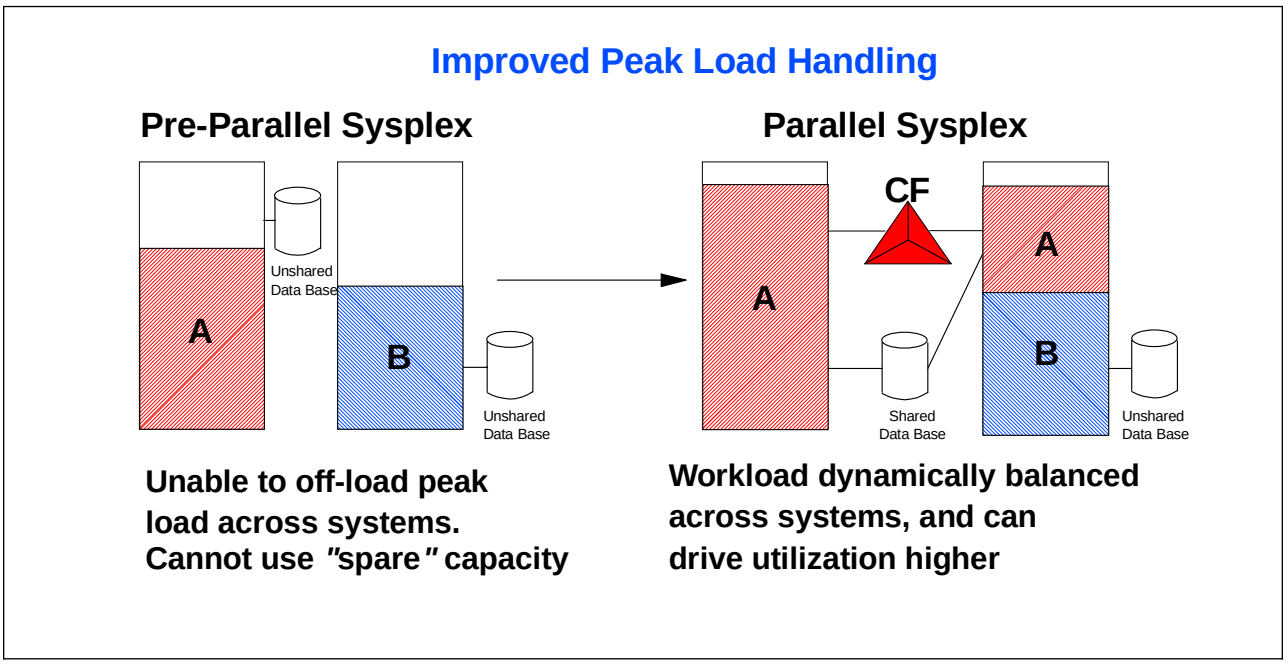


Figure 25. Handling Peak Loads with Parallel Sysplex

Figure 26 illustrates the ability of a Parallel Sysplex to continue to run an application even when a hardware or software failure causes the loss of a system (or subsystem).

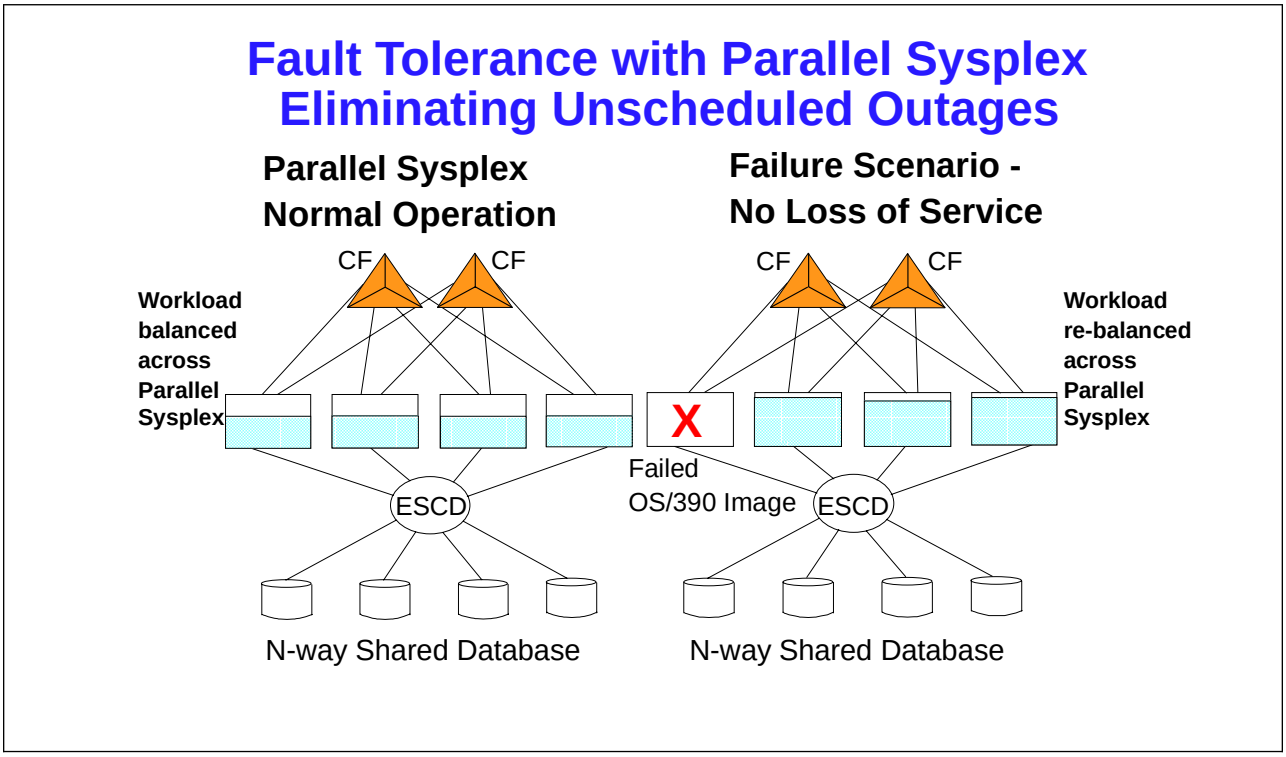


Figure 26. Continuous Availability During Unscheduled Outages

Figure 27 on page 68 illustrates the ability of a Parallel Sysplex to continue operation during periods when components are taken offline for maintenance or configuration change.

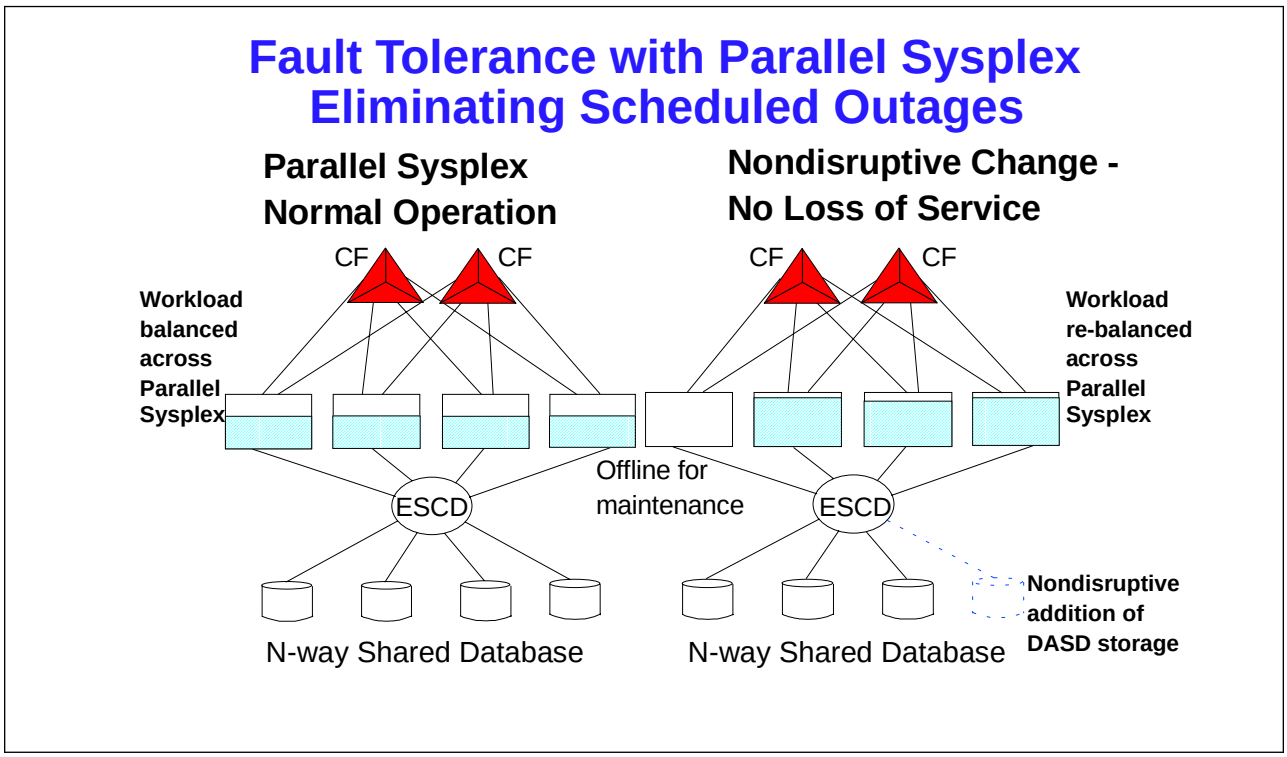


Figure 27. Continuous Availability During Scheduled Outages

Figure 28 illustrates the ability of a Parallel Sysplex to grow system capacity by adding new servers to the complex without interruption to service.

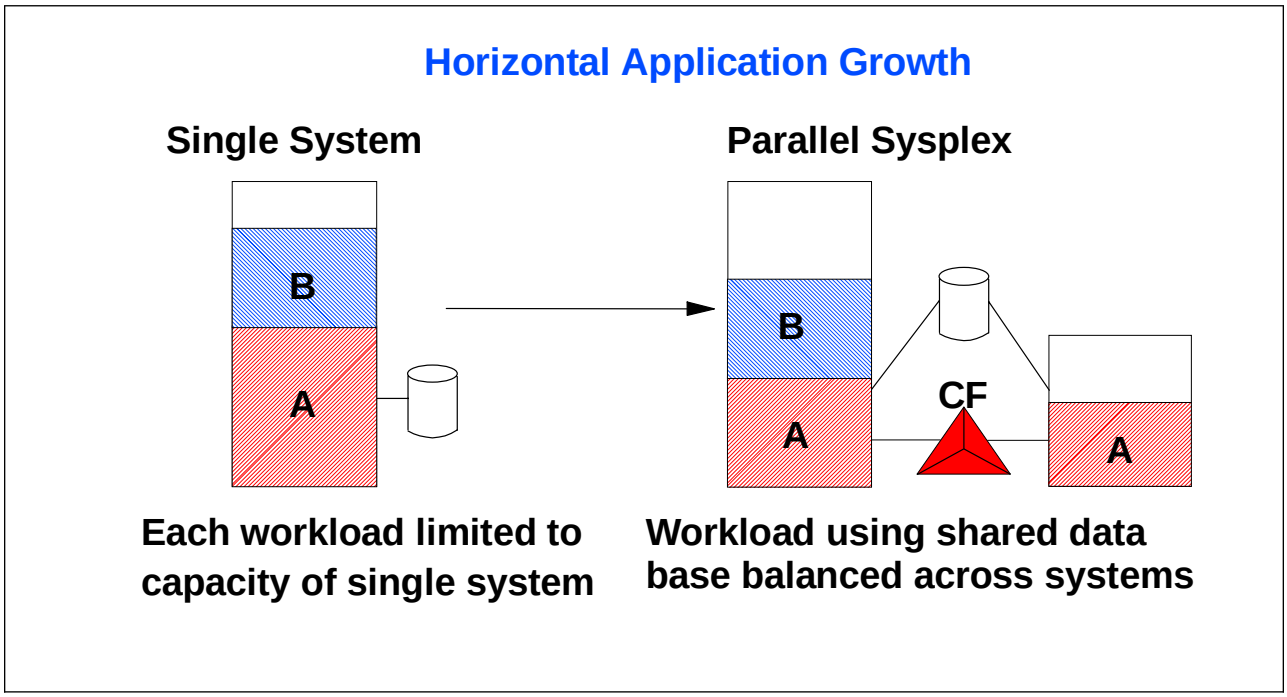


Figure 28. Handling Workload Growth with Parallel Sysplex

Chapter 4. Sizing and Performance Considerations

This chapter discusses the differences in performance characteristics between individual UNIX servers and a S/390 server running a consolidated workload.

Typical workload profiles are considered before comparing how the heritage of each environment has impacted the server design and how this affects performance.

Finally, we discuss sizing for S/390 and an approximate methodology for assessing the size of a S/390 for server consolidation.

4.1 Differences between UNIX Servers and S/390

In spite of the various marketing claims by the many vendors in today's marketplace, it is fair to say that similar levels of technology are used by all manufacturers for their servers offerings.

The base performance of the processor chips of any given generation are approximately the same no matter whether they are for IBM S/390, AS/400, RS/6000, or Netfinity, or for UNIX vendors such as Sun or HP.

The ability of a processor chip to perform raw calculations also does not appear to depend significantly on whether the instruction set is complex (CISC) or reduced (RISC).

The capacity to perform useful commercial work depends on other architectural and design choices and on the nature of the workload itself.

Traditionally, UNIX servers have used commodity packaging and support chips to keep costs low and run operating systems which are tuned to a specific task. As a result they have less internal bandwidth and trade off less of the CPU power for workload management, Reliability, Availability and Serviceability (RAS), integrity and so on.

This means that they can efficiently deliver high instruction rates as long as the workload does not overly stress either the movement of data around the system (for example, either a large working set or a large number of users), or does not have a need for complex workload management (for example, mixed workloads or high utilization).

IBM S/390 servers have used high technology packaging and more complex support chips and have traded off CPU power in the form of pathlength to provide high internal bandwidth, workload management, RAS, integrity, and so on. This means that they can safely and securely deliver service to many users running mixed workloads with acceptable and consistent response time.

The value of mixed versus single application consolidation is customer-dependent and determined by:

- Application
  - Portability
  - User count

  - Working set size
  - Scalability
  - CPU intensity
- Data flow
  - Batch windows
  - Networking in the transaction paths
- Customer background environment
  - Skills
  - Existing systems
  - Beliefs
  - Operational disciplines
- The value of service availability to the enterprise and the cost of outages

4.1.1 Application Profile and Operational Considerations on OS/390

In an S/390 Enterprise Server environment, generally a customer runs different application types on the same operating system. There is no dedicated machine for a specific application. Many applications can run concurrently on the same OS/390 system. Figure 29 shows a typical daily activity on a S/390 Server.

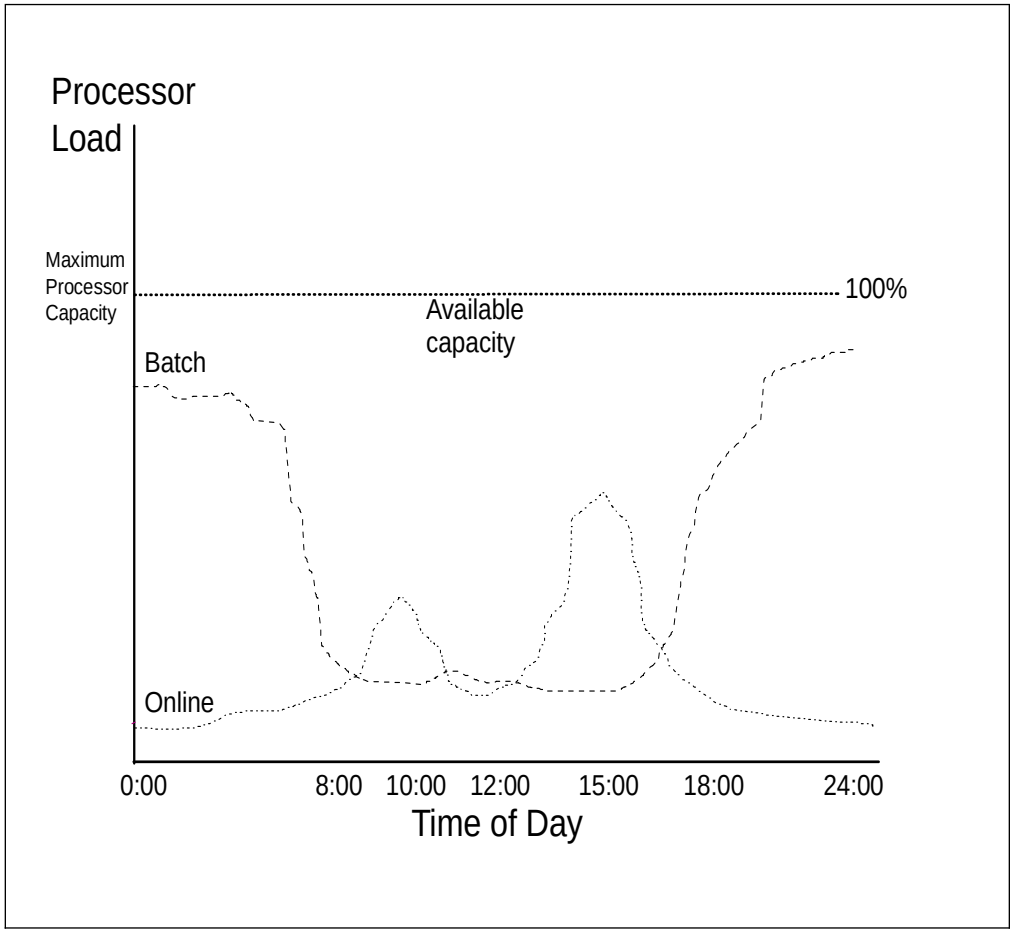


Figure 29. Typical Server Utilization for Online and Batch Workloads

Note that there are two workload types, *online* and *batch*. The online activity typically has two peak periods during the day, while batch activity mainly occurs

during the night. These two workloads can be easily merged on the same OS/390 server because the OS/390 workload manager can manage the resources (CPU, storage and I/O) according to customer policies defined to the OS/390 Workload Manager. Performance definitions or policies can be activated according to the fluctuating customer needs. For example, during the day, online activities can be prioritized, and batch activities can be given priority during the night.

Figure 29 on page 70 also shows that there may be spare capacity that could be used by a new application. This capacity is managed according to the OS/390 workload manager. In an OS/390 environment it is common to see a processor running at close to one hundred percent of its capacity, and still be providing good service to both online and batch workloads.

OS/390 also provides a secure, reliable environment for multiple concurrent applications by ensuring that no single application can “hog” resources, dominate the use of system resources, nor cause other applications to fail. Such an environment is mandatory when many servers and their applications are consolidated to a single system.

As discussed 4.3, “Sizing OS/390 UNIX System Services” on page 76, the additional pathlength required to implement this environment adds a complication to the process of trying to determine the capacity needs of a consolidated server.

What happens if the capacity of the consolidated server is exceeded?

With an OS/390 server, there are two options:

- Increase the number of processors in the SMP, and/or
- Cluster the servers in a Parallel Sysplex

In either case a S/390 solution offers excellent scalability. Figure 30 on page 72 shows the SMP ratio measured in standard IBM S/390 measurement cases (see also 4.2, “Sizing IBM S/390 Servers” on page 75). As the figure shows, the scalability varies depending on the workload type, with CPU-intensive workloads showing higher ratios than the more I/O-intensive workloads typical of commercial processing.

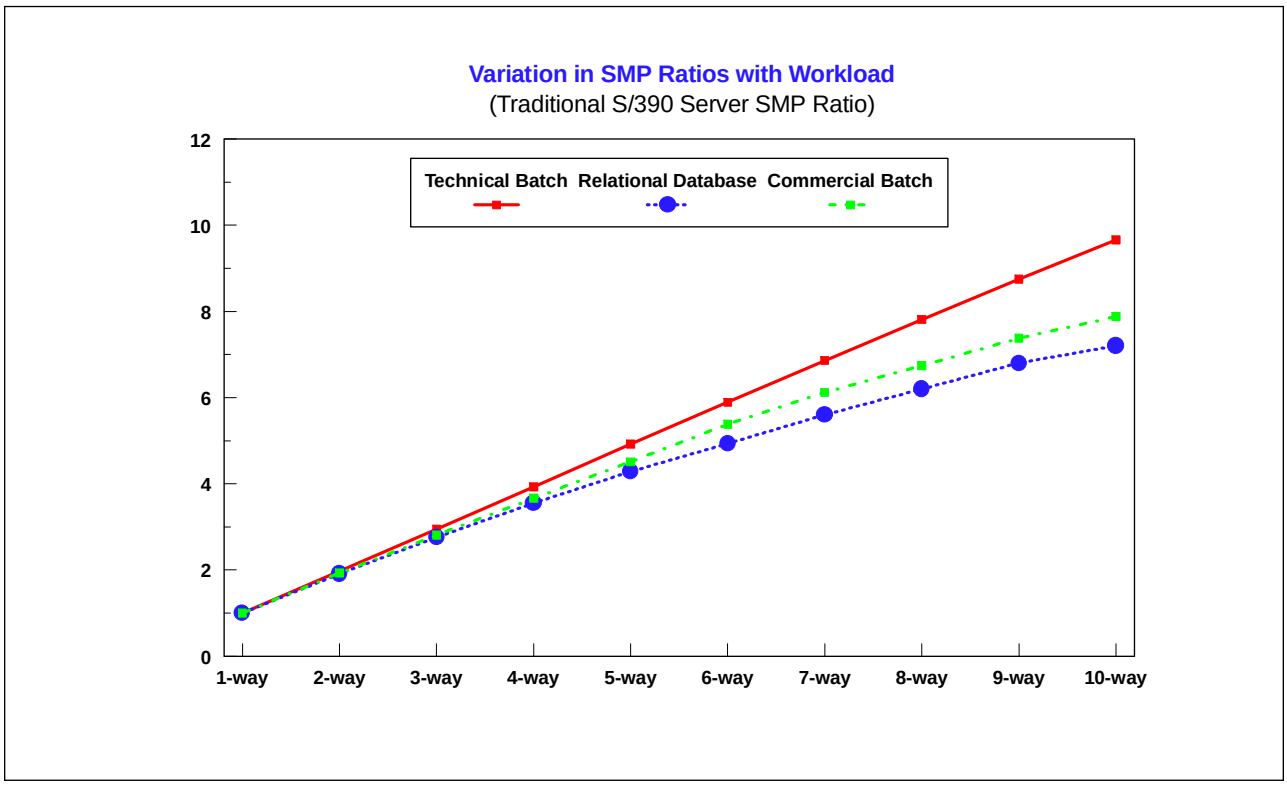


Figure 30. SMP Ratios for Different Workloads on IBM 9672 G4 Servers

The impact of SMP scaling on sizing considerations are discussed in 4.3, “Sizing OS/390 UNIX System Services” on page 76.

In 3.3.3, “Server Performance and Scalability” on page 41, the near-linear scalability of a S/390 Parallel Sysplex is discussed, together with IBM SMP scalability. In both environments, the excellent scalability is provided by the ability of all systems (engines within an SMP, or servers within a cluster) to access a single copy of each database.

This “shared-everything” approach should be contrasted to the “shared-nothing” approach which is universally used in UNIX systems.

Shared-nothing systems typically partition the database(s) across individual servers (or server engines within an SMP). When transaction arrival rates (workload skew) or database accesses (data skew) are uniformly spread across the database partitions, this approach can lead to good scalability and high transaction volumes.

In the vast majority of cases, such uniform distributions are only found in benchmark situations. In actual commercial workloads, workload or data skew results in additional overheads in inter-processor, or inter-server, communication in order to access database records in a different database partition.

As a result, while UNIX systems exhibit good benchmark performance, this is frequently not observed in running customer systems where, in order to maintain satisfactory performance, servers typically are only run at 50-60% utilization, and each server runs only one single application.

The S/390 shared-everything approach, while requiring higher overheads for single, simple applications, can drive multiple, concurrent applications to very high processor utilizations (90-100%), and can exhibit near-linear cluster scalability.

4.1.2 Application Profile and Operational Considerations on UNIX

The UNIX environment generally has a different approach, that of one application per system. This makes the system architecture both simpler and different, because each system is dedicated to one application and its database.

In this world, if the capacity of a server is exceeded the choices are:

- SMP growth
- Distributed databases with transactions and/or requests shipped between multiple servers
- Clusters of servers

In each option, the result is that the database is partitioned (for example, by engine within an SMP, or by server). The implications of a partitioned database are:

- Additional function/request shipping
- Additional inter-system communications
- Additional single points of failure
- Additional performance impact on scalability

This environment is difficult to tune and therefore often servers are unequally loaded and cannot be driven to maximum capacity.

Observations:

1. Most S/390 servers run at 90-100% utilization.
2. Most UNIX servers run at 40-60% utilization.

Some additional consequences of partitioned, unshared databases are how to satisfy growth, availability, and backup and recovery.

Considerations such as these have led to the sophisticated design of IBM's DB2 for OS/390 relational database product; see 4.1.3, “Database Considerations on OS/390.”

4.1.3 Database Considerations on OS/390

Following the database considerations in 4.1.1, “Application Profile and Operational Considerations on OS/390” on page 70, IBM chose to implement a shared-everything design for its S/390 cluster architecture, in order to solve customer demands for high volume, high growth with scalability, and high service availability.

As a result, it is possible to manage very large databases (terabyte size) with exceptional performance.

Using IBM's relational database systems, DB2 for OS/390, all data is shared and concurrently accessible by all servers in the cluster. Even though the database is fully shared by all servers, DB2 for OS/390 allows the database administrator to manage the database as if it were partitioned. This means that maintenance functions, such as database reorganization, can be performed independently on

each partition, resulting is higher availability of service and fewer scheduled service interruptions.

IBM today has a superior relational database management product called DB2 (the full name is DB2 for OS/390). DB2 has been improved for years in response to customers' needs. Business applications growth drives transaction volumes and DB2 for OS/390 scales well within an SMP. In addition near-linear scalable growth beyond the largest S/390 SMP is possible using IBM's Parallel Sysplex clustering architecture.

In addition to superior performance in OLTP processing, which is required when multiple smaller servers are consolidated to a single, larger server, DB2 for OS/390 on an IBM Parallel Sysplex cluster supports large data warehouse processing for Business Intelligence applications.

DB2 for OS/390 has evolved over the years by increasing the degree of parallelism in its database processing (see Figure 31).

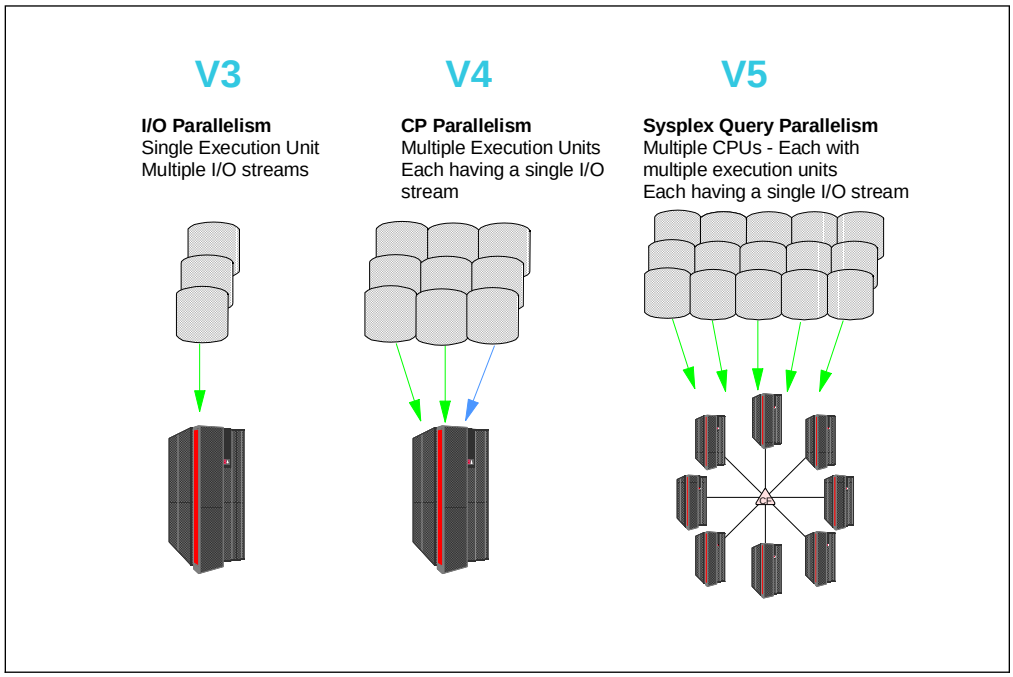


Figure 31. DB2 Evolution Steps

A good reason for server consolidation is to combine multiple instances of “data marts” into a single S/390 server in order to:

- Reduce the effort and time in moving data out of existing operational systems on OS/390 to the data warehouse and data marts for Business Intelligence processing.
- Improve the turnaround time for Business Intelligence application queries.
- Enable efficient processing of the wide spectrum of concurrent query types, including processor- and I/O-intensive queries, and short (a few seconds) to medium (a few minutes) to very long (hours) queries.

DB2 for OS/390 on a Parallel Sysplex near-linear (greater than 90%) scalability over a wide range of cluster sizes (see Figure 10 on page 43 and Figure 11 on page 43).

The Parallel Sysplex architecture allows the addition of up to 32 OS/390 servers in order to adapt the processor power the customer needs to face business growth.

The Parallel Sysplex architecture provides high service availability. If a processor fails, the remaining processors in a Parallel Sysplex still have access to the database because the database is shared.

4.2 Sizing IBM S/390 Servers

Vendors of UNIX (and NT) servers frequently make extensive use of industry-standard benchmarks in marketing their products. Generally such standard benchmarks either do not stress the I/O handling capability of the system (for example, tpc-A and SPECint), or only use a uniformly distributed arrival rate for transactions with a uniformly distributed data access pattern (for example, tpc-C). Neither environment is usual for commercial applications.

Such benchmarks enable the UNIX/NT vendors to perform extensive tuning, and even optimize their hardware and software designs to exploit the benchmark conditions.

However, as noted earlier in this chapter, real commercial systems do not exhibit such uniform patterns and usually do not achieve benchmark-like performance in production systems.

Because such benchmark systems are so tightly tuned, there is a strong dependence in the results on the particular combination of server architecture, UNIX operating system (vendor-dependent) and database system used.

In many cases, only a single benchmark number may be available for one combination of server model (SMP), version of UNIX, and database management system. Direct comparisons between other models in the same server range (for example, with a different number of engines in an SMP), or with other vendors' results may thus be extremely difficult, if not impossible, to do.

Thus while industry standard benchmarks may give a means of comparing the theoretical performance of different UNIX vendors, they are not very useful in assessing the capacity of a given system for a particular task (server consolidation, for example), nor do they enable easy comparison of different vendors' systems, or architectures.

S/390 servers, on the other hand, can be readily compared and sized in a realistic fashion by means of IBM-provided measurements.

The IBM Large System Performance Reference (LSPR) is a method designed to provide relative processor capacity for System/390 architecture servers.

The IBM S/390 Division runs LSPR benchmarks, which are controlled tests representing different workload environments (for example database processing, online transaction processing and batch). Each LSPR workload is a unique benchmark and is designed to be a closer match to actual customer environments than most industry-standard benchmarks (such as tpc-C, SPECint and so on).

In order to size a customer server requirement, the LSPR data that most closely resembles the customer workload is used. Data for all current IBM S/390 servers,

and for a large number of non-IBM S/390 architecture servers, is available. The best match to a customer's capacity needs can then be easily determined for either dedicated or mixed workloads.

In the S/390 server marketplace many people refer to MIPS (Millions of instructions per second) as a single-number metric for comparing servers from the same or different hardware vendors.

This metric represents simply the instruction execution rate and does not consider the internal processor design, or the ability of a server to deliver useful work for any given workload. It is particularly invalid for comparisons between servers of different architectures (such as IBM S/390, RS/6000, AS/400, or UNIX vendors such as Hewlett Packard or Sun Microsystems) and running different operating systems (such as OS/390, UNIX and NT).

IBM does not use MIPS as a metric for capacity planning because it is not sensitive to the type of work being processed. In fact the acronym MIPS is considered to stand for “Meaningless Indicator of Processor Speed.” However, the term is widely used and a single number per server can be established for a fixed workload mix and scale relative to a base machine, so long as the limitations of such a single number are understood by the user.

In fact, server capacity may vary significantly depending on the characteristics of the workload.

Production workloads may be batch-intensive, online intensive or a mixture of each. Workloads may also range from CPU-intensive through to I/O-intensive.

The LSPR measurements include seven different workloads representing many different customer workload types. Each individual LSPR workload is designed to focus on a single type of activity, such as interactive, online database, or batch.

By matching a customer's workloads to the LSPR workloads, a good understanding of the capacity of different IBM (and non-IBM S/390) servers for the consolidation can be obtained.

Thus, the challenge is now to understand the capacity and performance relationships between UNIX and S/390 servers.

4.3 Sizing OS/390 UNIX System Services

No single measurement can completely characterize the performance of either a S/390 or a UNIX server. Workload characteristics (processor or I/O-limited) and whether there are single or mixed applications determines whether benchmark results can be reproduced in a production environment.

Benchmarks for a processor-limited situation may give comparisons between processors which are in complete contradiction to benchmarks between the same two processors in an I/O-limited situation.

For example, for a SPECInt benchmark, an RS/6000 Model 59H is approximately three times the benchmark for a 9672 Model R12. For a multistream decision support benchmark, the 9672 R12 is approximately three times the RS/6000 59H in throughput (see Figure 32 on page 77).

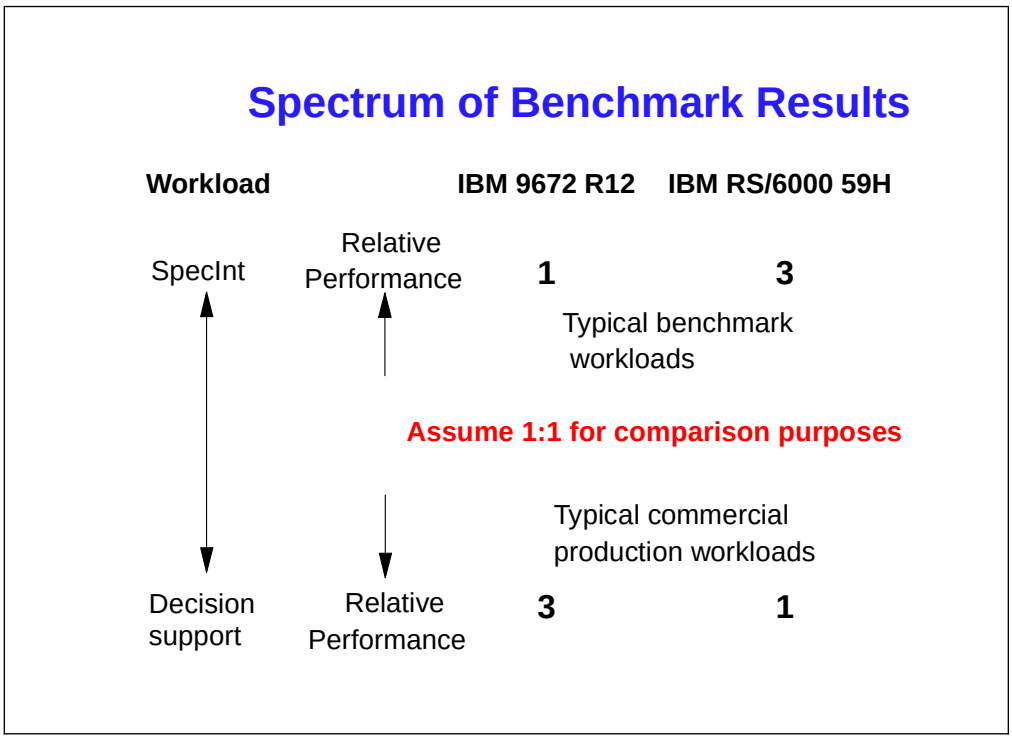


Figure 32. Spectrum of Benchmark Performance Comparisons

4.3.1 Industry Standard Benchmarks

The Transaction Processing Council (of which IBM is a member) has defined a number of standard benchmarks with the intention of enabling customers to compare the offerings from multiple vendors without the expense of running benchmarks of their own.

The tpc-C benchmark is the one most commonly used by UNIX and NT vendors when making their marketing claims. However, the tpc-C benchmark uses an idealized workload:

- There are only OLTP transactions, one bounded query and no batch.
- The workload utilizes resources at a constant “steady state” rate.
- The I/O is well-behaved and fairly uniform in size.
- Only 1% of the data is skewed so that there is 99% affinity in data, users and work.

Because of these factors there are no bottlenecks or hot spots caused by “user skew,” “workload skew” or “data skew.” Such factors are the norm in actual commercial systems, and even in the UNIX community, tpc-C numbers are only used as a starting point for vendor comparisons.

The following are some areas where industry-standard benchmarks can produce results inconsistent with what is observed in practice.

Resource Affinity: In most tpc-C benchmarks, the vendor, legitimately, will use system parameters to cause the workload (database and transactions) to be partitioned such that transactions to a given portion of the database will always run on the same engine on the SMP.

This means that cache contents, for example, are not polluted and large portions of the database can remain in fixed locations in memory.

These affinities result from the use of large private cache memories by UNIX servers. Because of the lower investment by suppliers in developing internal bandwidth, UNIX servers suffer more performance loss when the working set grows due to large numbers of users or when multiple applications cause cache misses.

To make matters worse, UNIX server caches tend to be deep and narrow[2] . While UNIX servers have big caches, data from each memory location has only one place to reside in the cache. Thus, if the CPU needs two pieces of information that use the same cache slot, the data must be moved to and from memory even if the rest of the cache is not being used.

Deep narrow cache arrays can be made from off-the-shelf parts. The resulting caches have good average performance but are more subject to "thrashing" (excessive non-productive rates of movement between memory and the cache), especially when non-sequential memory reference patterns occur. Having more than one program contending for cache space, or having many users on the machine, or an object broker making scattered references to objects in memory are all examples of situations which can cause thrashing. UNIX servers are therefore forced to choose between preserving cache contents and losing capacity to skew, or using any available processor and losing capacity to cache thrashing. All of these factors compound, creating pathological situations where UNIX server performance does not live up to benchmark results.

Providing multiple places for each memory location to map to within a cache uses special hardware, called Set Associative cache design, as is used in the IBM S/390. This design requires greater design effort and higher cost to manufacture. However, the benefit is that resource affinities do not exist, nor are they required in order to achieve high levels of performance in actual commercial processing.

Disk-to-channel and channels-to-processor affinities are also not present in the S/390 hardware or OS/390. These affinities result in queuing penalties when contention or "skew" occurs. The idealized workload found in tpc-C minimizes the skew, eliminating these penalties.

S/390 allows alternate pathing and dynamic I/O scheduling, which result in superior commercial I/O processing, but which also add overhead compared to the idealized benchmark.

Workload complexity: *Fragmentation* is the division of the resources among a large number of users. A large number of users are more likely to be using a wide range of resources (processors, cache, memory, I/O paths, devices, and so on), than a small number of users. Also the mix of resources used is likely to vary dynamically, so that any given resource is extremely unlikely to be dedicated to a small number of users. A workload with 1,000 concurrent active users is more fragmented than a workload with 100 concurrent active users run on the same machine.

---

2 For example, see pp 221-235 in *Configuration and Capacity Planning for Solaris Servers*, Brian L. Wong, Prentice Hall, February 1997 (ISBN 0-13-349952-9).

Fragmentation leads to reduced cache hit ratios and more dependence on high internal bandwidth. As the number of users increases, the speed of data movement becomes as important as processor power. That is, large user counts favor machines with high internal bandwidth. The IBM S/390 is still much faster than even the latest UNIX machines in this area.

For example, four L2 caches share a single 950 MB/sec bus in the SUN UE10000; up to 32 (but typically 16-20) L2 buses share a single 2.6 GB/sec bus in the Sun UE6000; whereas three L2 caches share two 2.6 GB/sec buses in the current S/390. This is because of the investment made in technology. The MCM packaging and power/cooling of S/390 results in more, wider connections between chips and shorter bus lengths. As a result, memory latency can be about half that of large UNIX servers and the aggregate cross-sectional bandwidth of the L2 Cache/Memory in the latest S/390 Enterprise Servers can be an order of magnitude higher[3].

Contentiousness: This results from competition for resources. S/390s are most effective at handling contentious, fragmented workloads, with large working sets, that tend towards being I/O-intensive. UNIX servers are best at parallel, homogeneous workloads which tend towards CPU intensity and small working sets. By way of example, a “light” OLTP application with less than 500 users and partitionable data (minimum skew) looks very good on UNIX. As the number of users goes over 750, the S/390 will look better. If the workload moves toward a larger working set such as in Business Intelligence, the S/390 would also improve in standing. When multiple query streams of ad-hoc queries are combined with OLTP, the S/390 will look extremely good.

A major difference in behavior between UNIX and OS/390 systems is shown in Figure 33 on page 80 and Figure 34 on page 81.

For single workloads the variation in response time with increasing processor utilization is very similar, with both systems being able to run at high utilization levels with acceptable response times. At low utilizations, UNIX systems will have slightly lower response times because OS/390 has code to handle mixed workloads and added functions which are not used by a single workload.

However, when mixed workloads are run on the same system, or there are large numbers of users, or when transaction arrival rates or data access are skewed, the profiles are quite different.

OS/390 can handle such workloads up to very high utilizations with acceptable response times, whereas UNIX systems start to give unacceptable response times at much lower utilizations.

---

3 For example, see pp 429-444 in *Shared-cache clusters in a system with a fully shared memory*, P.Mak et al. IBM Journal of Research and Development, Volume 41 Number 4/5.

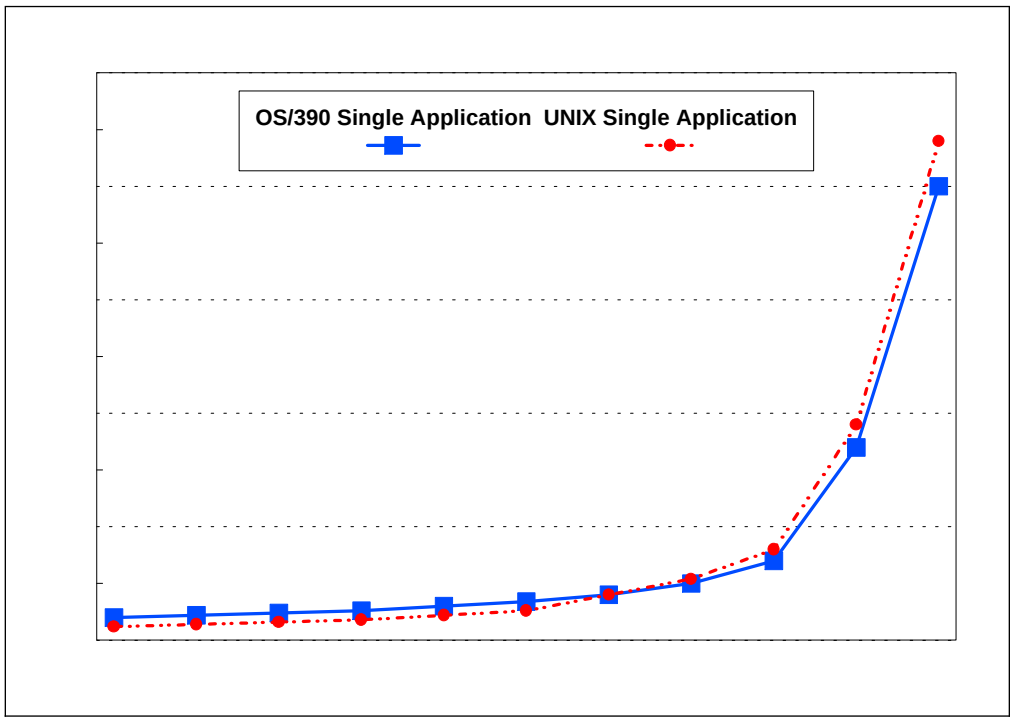


Figure 33. Performance Curves for UNIX and OS/390 for a Single Workload

The reasons for the differing behavior can be found in the differing design assumptions of OS/390 and UNIX.

As can be seen in Figure 35 on page 81, OS/390 evenly balances the use of the CPU (processor), processor bandwidth and workload management algorithms.

UNIX systems, on the other hand, have traditionally used the processor as a cheap resource and have not developed sophisticated designs for processor internal bandwidth or UNIX management of mixed workloads.

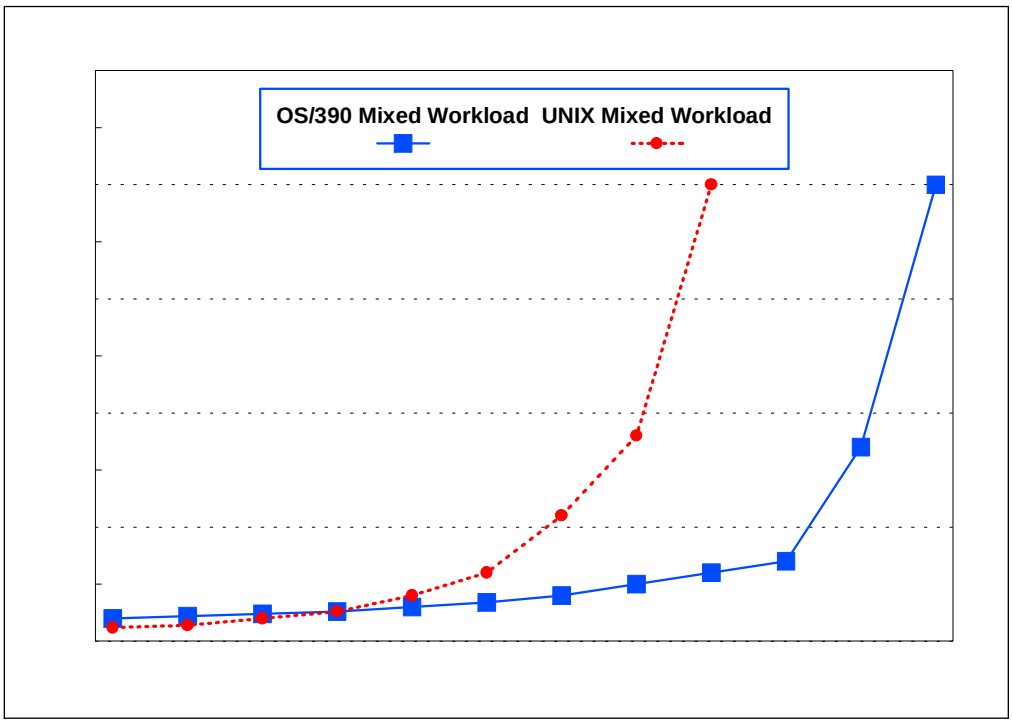


Figure 34. Performance Curves for UNIX and OS/390 for a Mixed Workload

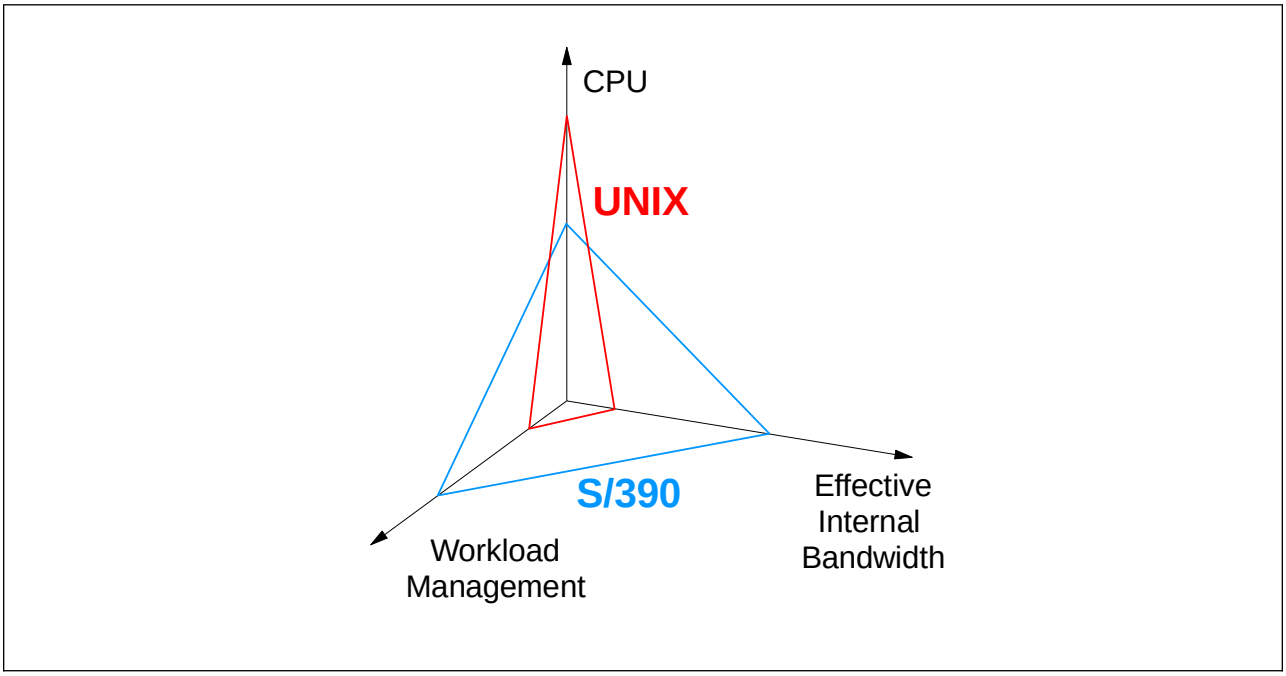


Figure 35. The Three Axes of Performance

The result is that UNIX sytems are very good at well-defined, processor-intensive single applications, whereas S/390 servers are better at running large, mixed workload systems.

4.4 Comparing UNIX Systems to S/390

As we observed earlier in 4.3, “Sizing OS/390 UNIX System Services” on page 76, comparing the capability of a S/90 server to consolidate the workload from multiple UNIX servers is very dependent on the nature of the workload.

However, the majority of commercial workloads appears to cluster in the lower portion of the range shown in Figure 32 on page 77, which gives a base from which the IBM S/390 division has developed a sizing tool (USSIZER - UNIX Services Sizer).

Since the use of this tool requires an understanding of benchmark processing, processor design characteristics, and the customer workload, and since it is continually evolving, this sizing tool is only being made available to trained IBM people.

4.4.1 Quick-Sizing a Server Consolidation

When a server consolidation is being considered, we recommend the following process:

1. Gather the following data for each server to be consolidated:
   - Average processor utilization
   - Workload characteristics
   - Server type - vendor, model, and so on.
2. Contact your IBM representative and arrange for the S/390 UNIX Services Sizer tool to be run. This will result in an initial sizing,
3. Perform a Business Solution Assessment (see Appendix A, “Consolidation Candidate Selection” on page 177) to determine the consolidation effort and benefits.
4. Then iteratively refine the server sizing as components of the server consolidation are ported and their performance assessed in the new environment.

Chapter 5. Customer Consolidation Example

This is an actual customer example of a UNIX application port from several UNIX servers to a single S/390 server. The customer is a large telecommunications company that had to develop a diversified portfolio of offerings in order to respond to the strong competition that it faced in the fast-growing US wireless marketplace. A decision was made by the customer to use UNIX, running on multiple servers, to develop and run a call and credit management application.

Although early design work supported this decision, it became apparent midway through the project that the platform would not be able to maintain the service levels necessary to support the business. Availability and scalability were the main concerns, although systems management and data synchronization were also seen as key issues to resolve.

These challenges convinced the customer that they needed to seek a more robust and scalable solution which would provide the highest levels of availability and reliability in the industry. After reviewing their various alternatives, the customer selected OS/390 running on a S/390 CMOS processor as their platform of choice.

The customer believes that the port would have cost the same if the application was moved to another traditional UNIX platform, or to OS/390. In addition, the development of new UNIX skills among the OS/390 staff was seen by management as a plus for the porting exercise.

5.1 Overview

The plan was to port the application to a S/390 CMOS processor and put it into production prior to integration into an existing Parallel Sysplex. A conversion of the database from Oracle to DB2 would also occur at the same time as the port from UNIX to OS/390 UNIX System Services. This database consversion was required because of the customer's differing standards for databases on UNIX (Oracle) and OS/390 (DB2).

This project required a varied number of skills and was large enough to require multiple companies and locations to be involved. Following is a list of these areas:

- IBM marketing and service
- IBM S/390 New Technology Center in Poughkeepsie, NY
- Other IBM S/390 people
- Other IBM divisions
- IBM System Migration Project Office (SMPO)
- A third party software vendor
- Another third party software vendor, who also acted as a consultant
- Two additional different consultants
- The customer

With all of these parties involved, it was very difficult to gather all of the people in one place to do the port. As a result the port was spread across separate locations:

1. Customer location in New York State - shell scripts and integration site
2. Chicago/Dallas/White Plains - Oracle to DB2 (IBM SMPO)
3. Waterloo, Ontario, Canada - C code (software vendor and consultant)
4. Poughkeepsie - C++ and Rogue Wave (S/390 Division)
5. Denver - vendor/consultant COBOL

This distribution of workload made the coordination of the porting dynamic, as well as somewhat disjointed. Each piece was ported and then transported to the customer site, where all of the pieces were put back together. That is when any architectural differences were identified and corrected.

5.1.1 People Days Spent on Project

Table 12 represents the total number of people days spent on porting the application OS/390. The days are broken out relative to the company that funded the people. These figures do not include the time spent by management. However, it does include the initial port and all subsequent retrofits and design modifications.

Table 12. Days Spent on Porting

Company	**Total days**
IBM	645
Customer	448
Rogue Wave	20
Grand Total	1,113

5.1.2 Application Overview

The application was designed to take raw data from telecommunications equipment, process it, and feed it into a data repository. This information would be processed in real time to provide a pro-active marketing function to the company, which would generate increased revenue by supporting sales to both existing and new customers.

5.1.2.1 Original Implementation

A schematic view of the application as it was running on the UNIX servers is shown in Figure 36 on page 85.

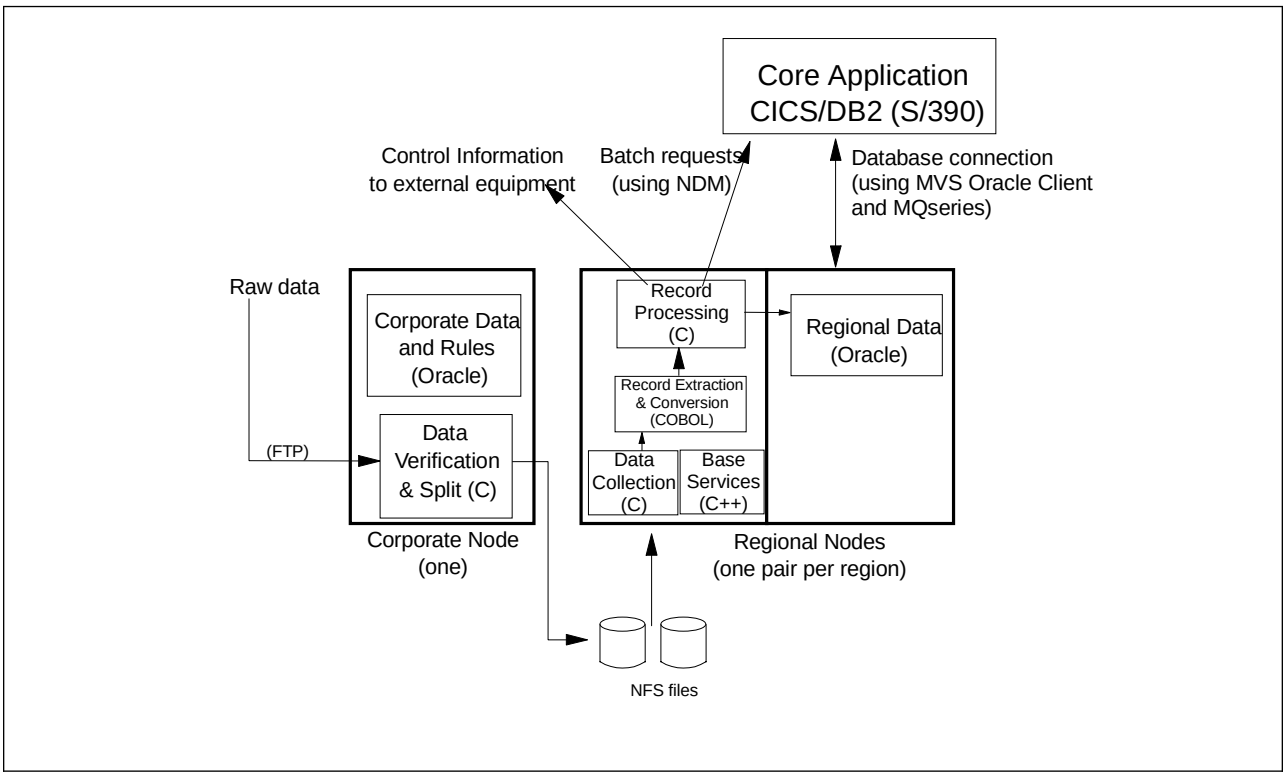


Figure 36. Original Multiple UNIX Application Structure

The raw data (effectively the number dialed and the length of call) is transmitted from external equipment via FTP over a TCP/IP network to one of the UNIX servers. Once received by the node, the data is checked for validity and then split into work queues, each queue corresponding to one of the company's marketing regions. These work queues, which are NFS-mounted, supply data to the nodes where the individual records are extracted and processed. Each record is stored in an Oracle database on another associated node. Depending on business rules, the record processing could also initiate a control command to the external telecommunication equipment. The two nodes are replicated for each marketing region.

The application also exchanges data with the company's core billing application (OS/390 CICS/DB2-based) to support enquiries and also maintain currency between the two databases used. The online data exchange was achieved through the use of Oracle Client/MVS and MQseries. The batch data exchange was carried out using shell scripts on the original system, which initiated batch jobs on the S/390 using Network Data Mover (NDM).

The application was written in a combination of Pro-C, C++, and COBOL as detailed in Figure 36 (the base services layer provides functions such as time management, data management, heart beat management and so on). The C and C++ modules are customized code built by Independent Software Vendors under direct contract. The COBOL routines were originally written for the host and modified, using Microfocus COBOL, to run on the UNIX servers.

An Oracle 7 database was used to hold the data, making extensive use of stored procedures which were written in PL/SQL. The core billing application ran on an ES/9000 9021-962 running MVS 5.2.2.

The system was operated manually with no automation present. The application design included a simple form of redundancy which is manually invoked. Typical recovery time for a system failure was approximately two hours.

5.1.2.2 Ported Implementation

The main structure of the application was preserved, with the data flows maintaining the same paths as before. The new schematic is shown in Figure 37.

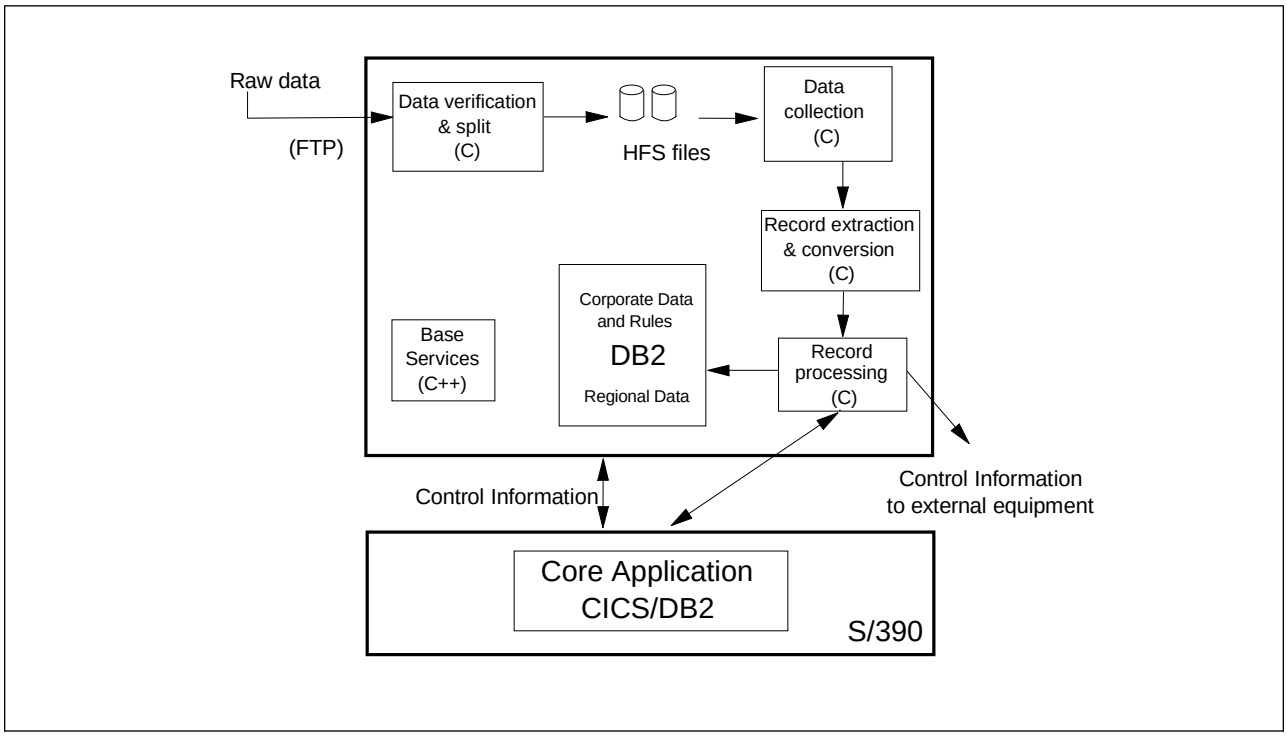


Figure 37. S/390 Application Structure

In this case, the data verification process continues to receive the raw data from the FTP link. Once processed, the data is placed into OS/390 HFS files. The NFS files were required on the original system to support access from multiple nodes, each of which ran a copy of the main application function. As the S/390 design has only one copy of the application, then only a single connection into the file system is required. This provides a more integrated design which will help in systems management and performance. Data is then collected from these files (in the same manner as in the original distributed UNIX design) and prepared for record extraction.

The COBOL program extracts and converts the data, then passes it in the same manner as on the UNIX servers to be processed and placed in the main database. Control signals are passed to the external equipment as before.

In the S/390 implementation, DB2 is used to maintain a single database environment on the S/390 platform, in contrast to the multiple Oracle 7 databases on the UNIX servers.

In the initial implementation, the data from the Oracle databases is held in a DB2 database that is separate from that used to hold the data for the core application. However. since a subset of the data in each database is the same (both databases are built from the same initial source), the intent is to merge the two

databases into one. This will remove duplication and support improved database management and design.

When converting from Oracle to DB2, the stored procedures were replaced with embedded SQL in new C routines. The MVS Oracle client function was replaced with the Distributed Data Facility (DDF).

The base platform services perform the same time and management functions that were performed in the UNIX server design.

A number of possible alternative approaches to the database migration were considered:

- Move Oracle to S/390 and then port to DB2
- Port the application to S/390 and communicate with Oracle via remote calls
- Move to DB2 first, then move the application

The last of these options was chosen (that is migration of the database from Oracle on UNIX to DB2 on UNIX, and then to DB2 on OS/390). This turned out to have been a poor choice, as it was later considered that moving the application to OS/390 using Oracle on OS/390, and then later to DB2 for OS/390 (the first option), would have introduced much less effort and delay into the port. However, there would have then been an additional cost for the Oracle on OS/390 software licence.

5.1.3 Management of Project

IBM believed that the opportunity to gain experience in the consolidation of a distributed UNIX application to a centralized OS/390 system would provide a valuable demonstration of the benefits of S/390.

The customer also saw business benefits from the consolidation and was keen to work with IBM and others in order to complete the project.

For these reasons, IBM made an investment in the customer's project and the New Technology Center in Poughkeepsie committed to support the port, while the customer purchased an IBM 9672 R22 server.

The New Technology Center agreed to provide one person to manage the IBM side of the port. Other groups within IBM agreed to provide funding to port some portions of the application. IBM's Software Migration Project Office (SMPO) agreed to convert the Oracle database to DB2 on the 9672. The consultants that developed the application were contracted to provide application design and COBOL support.

In the enthusiasm of the initial stages of the project, a number of areas were neglected and led to later delays in the project. These included:

1. No project manager was initially assigned.

   The project start-up was a team effort, whereby all of the involved organizations met and formulated a plan. Each party assumed that the others were providing the project manager. However, what was initially in place was:

   - A customer project manager, who was responsible for interfacing to the application port manager, resolving global issues internal to the customer, approving use of outside resources, and reporting status to management.

- An IBM project manager, who was responsible for providing an interface to IBM OS/390 UNIX System Services development, coordinating IBM support, expediting problem resolution, and interfacing with the the customer's project manager.

It quickly became apparent that such a division of responsibilities was unworkable, and it was agreed that IBM would provide a project manager. Eventually, a project manager from the IBM S/390 division was put in place, but a two- to three-week slip in the schedule resulted.

2. Inadequate planning for the total port.

   All parties were so anxious to get an application consolidated to OS/390 UNIX System Services that nobody took the time to take a good look at the configuration and the application in order to produce a comprehensive plan for the port. All of the parties, except for one consultant, had no idea of the scope of the job in which they were becoming involved. This was particularly true with the approach taken to the conversion of the database, and this also contributed to delays in the project.

3. No definition of the scope of the port.

   There was no documented scope of the port. What was actually provided for porting was a version of the application that was still in test. What was finally delivered to the customer was a version ready to put into production, including new functions required by the business that were not in the original system submitted for consolidation.

   The port required redesign of many functions because of the changes introduced by having a single system versus a multi-node system.

4. Incomplete plans for the Oracle-to-DB2 conversion.

   The Oracle-to-DB2 port deserves mention because of the huge discrepancy between the initial estimation that the conversion would be complete in two months and what actually transpired.

   The original estimator's plan turned out to be impractical in that his plan was to provide the customer with code that had been only successfully pre-compiled and compiled on a UNIX version of DB2. This did not include the additional tasks of compiling the code on the S/390 and functional verification of any of the SQL or C code on the target system.

   In addition, a large number of field changes were required in order to conform to the customer's database standards for the S/390 environment (which differed from those for Oracle on UNIX).

Overall, the lack of an agreement defining the roles and responsibilities of all parties involved caused at least a 4-to-6 week delay in the schedule. If the project manager had been appointed with the appropriate role definition up front, the first activity would have been a sizing estimate. An up-front review of the design would have discovered all of the functions and facilities that were required in order to port. This would have avoided a large amount of the design and “fix-on-the-fly” activity that actually occurred.

5.2 Customer Benefits

The benefits of consolidating the various UNIX servers onto S/390 were perceived by the customer to be as follows:

- Improved application Management

  The UNIX environment employed three people whose task it was to query the status of the various work queues as this was the only indicator that the application was actually running.

  In the new S/390 enviroment, standard OS/390 monitoring tools made it easy to determine the application status.

- Better scalability

  Application growth, driven by business volumes, could be accommodated at a lower cost than with the UNIX servers because of the better scalability of the S/390 server.

- Improved availability

  Improved application availability was one of the criteria which led to the decision to consolidate on S/390. Improved availability was in fact achieved in the new environment.

- Fewer support staff

  Since the customer already had a S/390 operational environment, the consolidated application could be supported with no change in existing S/390 support staff.

- Reduced number of databases to support

  The application running on the UNIX servers required two different databases: Oracle on UNIX and DB2 on S/390. Following the consolidation, support was only required for DB2 with a consequent saving in support people and costs.

5.3 General Observations

The following subsections summarize some of the observations made by members of the IBM project team on their experiences of this project.

5.3.1 Customer Involvement in Project

- The sponsor for the project was the Chief Information Officer (CIO). Approximately 6 months into the project, a new CIO was appointed.
- The customer project manager was assigned from the application development area.
- No organization changes were planned or occurred as part of the project.
- The networking specialist and the systems programmer had no education in S/390 or OS/390 before the project commenced.

5.3.2 Lessons Learned

The following are the key lessons learned from the porting experience:

- It is very important to do up-front planning.
- It is important that the same people who wrote original code are also used in the porting process.

- All Independent Software Vendor (ISV) applications must be checked to ensure that they do not also need to be changed.
- Education prior to the porting activity is essential in order to save time learning on the job:
  - Management should have high-level education on the differences between OS/390 and UNIX from their perspective.
  - Coders should have an appreciation course to understand the high-level basics of OS/390 and things that are essential to know.
  - System programmers, network specialists and storage and security administrators should have an appreciation course to understand the high-level basics of UNIX and things that are essential to know.
- A staged implementation should be considered.
- If a series of ports are to be carried out, then a small core team should be established and work on each port. Skills in this group should include application architecture/design, UNIX administration, and OS/390 system programming.
- The people doing the work should attend *all* meetings even if they are very busy or not directly involved in the subject of the meeting. This ensures that all persons involved are continually aware of the the project status, and understand the significance of all areas of code change (and to veto or modify these if there is any project or code impact that the proposer of the change did not anticipate).
- There should be a communications channel between UNIX and OS/390 technicians, in order to ensure that the requirements of both environments are understood by both groups.
- In this project, performance testing was carried out on small sections of the application. A better approach would be to run complete end-to-end measurements.

5.3.3 Factors that Affected the Project

Some of the factors that had a significant, negative impact on the migration were:

- The scope of the project was not fully understood at the start of the exercise.
- The customer's knowledge of the application was not extensive.
- Oracle was ported to DB on the UNIX server, and then to S/390. In retrospect, it would have been quicker, easier and less error-prone to have ported the application from Oracle on the UNIX servers to Oracle on OS/390 UNIX System Services, and then converted to DB2 for OS/390 after the application was running cleanly under Oracle on OS/390.
- The application was designed by external consultants, who were not available during the actual porting process.
- Major sections of code were contracted out to other consultants, who did not support the port with great enthusiasm.
- Under-resourcing by all parties extended the time-scales. This brought the port into a critical business period, which delayed the cutover to production.
- Some changes had to be retro-fitted up to three times as the team attempted to respond to requirements from the business. As in all large projects, the application design should have been frozen during the porting.
- It was difficult to keep the project under control due to the geographic and management diversity previously mentioned.
- There were a number of issues with automation (starting and stopping jobs, processes and so on).

- The installation of the file system was straightforward, but the HFS setup was complex. The system files were set up by the OS/390 system programmers, while application files were set up by the storage management group.
- MVS/ESA 5.2.2 and associated products were used initially, and this installation took up most of the principal system programmer's time. OS/390 V1 R2 was introduced later in the migration because some of its features were required for the port. OS/390 integrates many previously separate products, and consequently this installation effort would have been significantly reduced if OS/390 had been used from the start.

5.3.4 People Considerations

As in all major projects, the experience and training of the people staffing the project can have a significant impact on the progress of the porting project, as illustrated in the following list:

- The philosophy of UNIX needs a different mind set from that of the OS/390 user. For example, the rebooting of a system to recover from a problem is a culture foreign to OS/390 staff.
- The OS/390 systems programming group needs UNIX skills, and UNIX-trained staff need education on OS/390. For example, one systems programmer said it would have taken three months to install the system with no prior experience, but that this would be reduced by 50% if the proper education had taken place.

  With this lack of experience in both environments, it was sometimes difficult to find required information (for example, how to set up Syslog daemons under OS/390 UNIX System Services).
- IBM forums (both internal and Internet-based) should be used to find information and get answers to queries.
- TCP/IP skills are very important when porting between such different environments. The ITSO Redbook *Accessing OpenEdition from the Internet*, SG24-4721 is an extremely useful source for answering many TCP/IP questions.
- The RACF coordinator has to take on UNIX responsibilities when the system goes into production. The differences between security in UNIX and RACF in the OS/390 world need to be carefully considered.
- Differences between the two environments that application developers would notice are:
  - The shell and debugger are different.
  - The ability to view UNIX processes is restricted by RACF.
  - The inability to see a dismounted home directory under OS/390 UNIX System Services - the only way is to use the directory name to remount it.
  - There is no NFS file locking under OS/390.
  - Failure codes can more difficult to research if there is a lack of familiarity with OS/390 documentation.
  - Change management disciplines may be different.. For example, UNIX developers often carry out a fix, test it, then apply it to the production system without following any formal procedures. This lack of change management is alien to the OS/390 world.
  - With OS/390, modification of the operating system is under the system programmer's control (according to RACF authorization), and users no longer have this ability as they have under UNIX.
  - There are differences in compiler options and usage.
  - There are differences in queuing mechanisms between UNIX and OS/390.

5.3.5 Technology Considerations

Some observations of the differences in the two technologies were:

- There is no support for X-windows or Motif under C++ on OS/390.
- OCS (on RS/6000) was used to provide RLogin.
- NFS locking does not exist on OS/390 UNIX System Services,
- ASCII-to-EBCDIC conversion caused some problems in application porting.
- A 3172 connection was used into IP networks.
- Hooking TCP/IP into OS/390 was not easy. Extensive use was made of *Accessing OpenEdition from the Internet*, SG24-4721.
- The kill -9 command would not terminate all processes in OS/390. MQM connections would be in a wait state and would not receive the kill command until they were re-activated.
- The Vanguard front-end RACF tool did not support OS/390.

5.3.6 Operations

A more complete discussion of the operational considerations involved in a consolidation from distributed UNIX servers to OS/390 can be found in Chapter 9, “Operational Considerations” on page 125.

Some specific observations from the customer's experience are:

- The operational philosophy of the UNIX-trained operators was very different from that of centralized OS/390 operators and that led to misunderstandings of the new environment.
- It can be complex to manage a mix of OS/390 UNIX System Services and other UNIX systems.
- The ability to have a single common view of data in the consolidated environment was seen as a major advantage later in the project.
- The maintenance philosophy changes when OS/390 UNIX System Services is introduced, due to the lack of file-sharing in OS/390 UNIX System Services.
- There were major differences in the tools used for change management. SCLM was used on OS/390, while SCSS was used on UNIX; this resulted in a very poor change management system.

5.4 Porting the Pieces

The following subsections discuss some of the issues that arose in porting the various components of the original customer application.

5.4.1 Platform Services (C++)

An early assessment by one consultant indicated that the C++ port would be fairly easy. The work would consist of getting the code from the customer, getting a clean compile on OS/390, and then working with the rest of the customer project team to integrate the C, C++, COBOL and DB2 ports on a system at the customer site. The consultant also believed that the most expeditious method of porting this code would be to move it to OS/390 UNIX System Services, do the compile, and then fix the errors. At that time it was estimated that there was between 50 and 60 thousand lines of C++ code. The C++ porting tasks that were defined at a subsequent project meeting were:

- Modify code to clean compile

  - Developer orientation - The developers of the middleware will spend two to three days reviewing the architecture and functionality of the code.
  - Identify system-dependent APIs.
  - Validate identified APIs.
  - Socket data stream translation conversion.
  - Determine how to process the UI database (port or convert).
  - Modify platform tables as needed.
  - Platform functions to port
    - Heartbeat manager
    - Process manager
    - Timer manager
    - Data manager
    - Statistical measurement
    - Message processing
  - Review message text and adjust to handle conversion.
  - Port the test emulator - This is a test vehicle used to test the middle-ware.
  - Port stress test module.
  - Port and validate the Buildall script.
  - Document the port and identify any areas where performance enhancements could be made.

A subsequent assessment by a consultant highlighted some additional areas of work:

- A tools class library from Rogue Wave required porting.
- A fork/exec of an xterm in WinClass.C, used to display a log file, required reworking to adapt to the OS/390 UNIX System Services environment.

The resource to perform the port was difficult to find. The consultant could not support a C++ port, and there were no skills available with the customer or within IBM. After much searching, an external resource was employed to analyze the scope of the project and perform the compiles, which added a delay of several weeks to the project.

Once the port was underway, various design considerations were identified. The first was the dependency on XWindows and Motif. The number of modules that made calls to these services was much larger than initially thought. Secondly, MQSeries was used to provide cross-system message queue communications with the CICS/DB2 core business application.

There were major delays in the port due to the length of time it took to get problems resolved. Some of the problems that were encountered are:

- There was no XWindows support for C++ a (UNIX user interface was used instead).
- There was compiler/prelinker failure.
- DBX does not work with C++ code.
- There was no NFS locking for HFS files (a locking process was written).

Some of the problems were bypassed, which prevented some delay, but that did not avoid schedule slips caused by the first instance of the problem. For instance, the DBX problem caused the debugging of the ported code to take an estimated four times as long as it would have if DBX had worked.

5.4.2 C Code

The port of the C code was performed by a consultant with a team in Waterloo, Canada. Their extensive experience highlighted that a comprehensive porting plan was critical to success. They also installed the ported code on the customer's site.

It was necessary to make some changes to the function provided by the C modules. Most of these were due to the change from a multi-node to a single node environment. Although it was believed that the modules should have executed in OS/390 UNIX System Services, several problems were encountered when fitting the code into the production environment due to the mixing of the data from different marketing divisions.

In the original UNIX server system, each market was processed on a node (or set of nodes) that was dedicated to a single market. In the OS/390 UNIX System Services environment, there is only one node and all of the data was in one database. This had the potential of causing problems when multi-market switch data was processed.

5.4.3 COBOL

There were two aspects of the port that related to COBOL. There were some COBOL routines which run on the CICS/DB2 billing system which required modifications to support the database conversion (the previous implementation used the OS/390 client code provided by Oracle to call stored procedures).

Another COBOL program, written in Micro Focus COBOL, required porting from UNIX to OS/390 UNIX System Services. The program, which was originally written as part of the host CICS/DB2 core billing application, had been ported from OS/390 COBOL during the original application development design for the UNIX system. So it was considered easier to take the OS/390 production version of the code and make updates to it to add the function that was added to the UNIX version of the program. One reason for this decision was due to a poor implementation of a C subroutine that was used to allocate the output files in the Micro Focus version. The OS/390 version had an assembler routine that allocated the files using dynamic allocation.

The easiest method to integrate this program into the host environment was to execute it as an OS/390 batch job. The shell script that previously called it in the UNIX server environment had to be split into two pieces. Originally the shell script called the COBOL program and knew where the output files would be placed. When it was executed as an OS/390 batch job, the file names were OS/390 files. These file names had to be passed back into the shell environment after the COBOL program completed.

Three areas were identified that had to be updated due to the move from a multi-node environment to the OS/390 UNIX System Services environment. The first two were the ability to run multiple copies of this program and to provide a unique identifier for the dynamically allocated files. It was also considered desirable to be able to link the output to the batch job files in the System Display and Search Facility (SDSF). This required a unique jobname for each job, as well as an associated identifier for the output files. The shell script that submitted the job was modified to call a routine which would pass a batch sequence number that was used in the jobname and in the dataset name(s) created by dynamic allocation. With unique jobnames and an assigned job class, the number of concurrently

running copies of this job could be varied by starting and stopping initiators for that class. With all of the outputs containing the unique sequence number, it was possible to provide a means of tying together everything associated with the job.

In the original system, the names of the output files were placed in a directory where the shell script would look for it. On the ported implementation the COBOL code submits a batch job for each of the files created. Then Job Control Language (JCL) is used to start a shell script from a batch job. The name of the file to be processed into the shell script is passed as a parameter. This provided a timely solution to the back end of the COBOL processing.

5.5 The Integration Plan

The plan for integrating all the converted components of the application onto the single OS/390 system was to divide the process into two phases:

Phase 1: This phase consisted of initializing the file systems and the version control system, and verifying the functions needed by the integration team.

Phase 2: During this phase all teams worked together to install the ported code and verify the functionality of the entire application.

5.5.1 Comments on the Integration Plan

The plan, as described, was an idealistic approach to integrating the application. As it turned out, the actual integration went nothing like the plan. Problems were encountered in many areas. The ones that caused the largest delays were:

- As previously mentioned, DBX did not work for C++. This meant that debugging took an estimated four times as long as it would have if DBX had functioned properly.
- There was no Xwindows support on OS/390 UNIX System Services for C++. This forced the use of User Interface (UI) on a UNIX system. This also meant that an ASCII-to-EBCDIC conversion was required on all messages sent between the various processes and the UI. In addition, the UI database had to be NFS-mounted by the UI and stored in EBCDIC. Conversion of the contents of the database was not a global convert because the records in the database were a combination of text and binary data.
- NFS on OS/390 did not provide locking, so code had to be developed to implement a locking function for database updates by the UI. These caused extensive delays in the port of the C++ application, which overran into the integration phases.
- Design changes were required for the process flow because of the former multi-node topology.
- Additional COBOL changes were required in order to compensate for functions that were previously provided in Oracle as well as the multi-node architecture.
- OS/390 V1 R2 was required in order to solve some problems. There were problems like DBX functioning and Language Environment upgrades that conflicted with migration to OS/390. It was decided to migrate to OS/390 instead of taking time to apply the massive fixes and upgrades to MVS 5.2.2 that would otherwise have been required.

- Lack of resources. The actual integration team consisted of three programmers and the project manager. The integration manager was assigned late in the cycle and was not committed full time until after integration started. No one on the team or even the combined team knew the entire application. This caused several delays for doing research to determine how things were supposed to work.

  In addition to the four primary members of the team, additional support staff were required. These varied from DB2 database administrators to networking configuration and automation specialists. Each of these resources had other assignments and often other priorities when it came to performing tasks related to the port. Many times this resulted in delays while waiting for tasks to be completed.

- It was a new world! Everyone on the support team as well as the porting/integration team was learning. Not only was the world of OS/390 UNIX System Services new, but for the majority of the team, it was the first time that they had seen the application that they were porting.

5.5.2 Actual Integration

All of these factors extended what was planned to be a two- to three- week integration phase to be a three-month effort. Details of the problems and inhibitors can be found in the respective port description sections. In hindsight, a lot of time could have been saved had the project manager been more familiar with the application and its design. Many of the changes that were required after the code was ported could have been made during the port if the team that was performing the port understood the architecture and design of the entire application. As it was, no person with that level of knowledge was available until an integration manager was assigned full time. People were available for questions, but no one monitored the port from the overall application perspective. The team did not even work in the same location. This made simple questions that normally would be resolved by walking down the aisle to another's office turn into a long-drawn out stream of notes between the parties involved. Often these were funneled through the project manager. The project manager would chase down the answer and feed it back to the requester. As could be expected in this type of communication, details were lost in interpretation and minor discrepancies were not resolved because of the complexity of communication.

5.6 Summary Analysis

The port took much longer than planned. Contributing to this was a long list of items that individually did not appear to be way out of line, but which in total resulted in the elongation of the project.

A number of these items deserve mention:

- The lack of a statement of work, a Document of Understanding, or any other description of the port agreement caused a 4-to-6 week delay.
- The lack of C++ skills caused a delay of three weeks.
- Inadequate design analysis caused the schedule to be extended by six weeks. Many aspects of the design of the application were based on a multi-node architecture and had to be modified when moved to a single-node environment.

- The installation of OS/390 V1 R2 resolved problems that had caused a two-week delay. Once OS/390 V1 R2 was installed, only three problems were experienced related to OS/390 UNIX System Services, and only one of them impacted the testing (and that was resolved by a fix that was already available). The impact to the schedule was one day.
- The learning curve added approximately 25% to the schedule. This affected the people doing the port, as well as all of the system support people. Over 21 products were used in the application and the port. No one on the team had knowledge in all of them; most had to learn the products as they encountered them.
- The lack of application expertise caused approximately two weeks of additional effort.
- The addition/redesign of function caused a two-week extension. It is very difficult to restrain someone from making improvements to code when porting an application. We had to put a freeze on making any changes that were not needed to complete the port in order to stabilize the code for function testing.
- Changes to the structure of the database when moved from Oracle to DB2 caused one-week delay. This delay occurred during function testing because until the application used the redefined fields, the problem was not detectable.

These delays overlapped and occurred in various areas, so they cannot be added or subtracted from the schedule. The net result is that it appears that in total there was between four and and six months of “slip” in the initial schedule. With a team of experienced people, this port should have been possible in four to five months. That would have made the port complete with only a two-month extension of the schedule.

It is also recommended that a port should be a phased approach. Porting the application should not be done at the same time as a database conversion. Many problem resolution times were extended due to the difficulties in establishing the origin of a problem, the choices being the data in the database, or in the new SQL implementation, or in the application. For this application, either Oracle should have been used on the OS/390 system or the application should have been converted to DB2 on the UNIX server, with the field changes that are required on the host DB2 system being integrated prior to the port.

What could have been done to avoid the slips in the schedule? Actions that could have been taken to improve project elapsed time are:

1. Perform an assessment of the application to be ported (see 2.5, “Is My Application a Good Candidate for Server Consolidation?” on page 28 and Appendix A, “Consolidation Candidate Selection” on page 177). This assessment should provide a set of tasks and sub-projects that need to be completed. These, in turn, need to be prioritized and sized.
2. Identify the skills required to complete each task.
3. Identify the data required to complete each task.
4. Produce an agreement that identifies the roles and responsibilities of each of the companies involved.
5. Identify the financial responsibilities of each organization involved with the consolidation.

6. Identify the architectural changes required because of differences in server platforms.
7. Identify all external dependencies, such as class libraries, products, and vendor support.
8. Formal education for the porting team is required on the S/390 and OS/390 UNIX System Services in order to reduce the learning curve.
9. Freeze the application code until the port is complete. Do not allow any changes that are not required to make the application run on the target system. Performance and complexity reduction changes can be completed after the port is complete.
10. Move the database without change. Any changes required in order to conform to company standards should be done *after* the move to the new platform.

Part 2. Guidance in Consolidation of UNIX Systems to S/390

Chapter 6. Project Management and Education Considerations

The success of any project is heavily dependent on the skills of the people staffing it. Therefore, in this chapter we discuss the importance of both project management and education considerations.

6.1 Project Management Requirements

We discuss project management for server consolidation under the following headings:

- Project Categories
- Planning
- Alignment of Objectives
- Project Sponsorship
- Culture - OS/390 vs UNIX
- Communication
- Geography
- Change Control
- Workers Should Attend Meetings
- Staged Approach
- Critical Resources
- Migration Plan

6.1.1 Project Categories

We can distinguish two major categories of server consolidation environments:

1. Customer code porting project with customer code port, database migration and Independent Software Vendor (ISV) code.

   Since each customer application has its own unique combination of language, coding experience and standards, database design and ISV products, the project management for this type of customer is likely to be complex. The remainder of this chapter is largely targeted at this customer set.

2. Independent Software Vendor (ISV) porting project.

   This type of project may be much simpler to manage. Usually an ISV has good business reasons to port his application on OS/390, based on an identified prospect. However, many of the ISVs do not have any in-house S/390 system or skills.

   IBM has established a number of porting centers worldwide. These centers provide an environment (hardware, software, access to skills) in which a UNIX programmer can conduct a port.

   This is usually a three-step project:

   a. Feasibility studies leading to technical options
   b. Product integration onto the target platform
   c. Product packaging

   The conclusions after each step should be formally documented and agreed upon.

The whole procedure should be done in close relationship to the prospect whose requirements originated the project.

6.1.2 Planning

As with any other project, a porting project needs a clear definition of all the tasks to be done during all the project phases. This must be done based on the objectives of the project and according to a calendar determined by the customer, and validated by the customer and an experienced project manager. Use of the Business Solution Assessment methodology (see Appendix A, “Consolidation Candidate Selection” on page 177) can be of value with this effort.

All the phases should be defined and the scope specified. The deliverables should be identified and validated by all parties. Some of the elements may be conditional on the start of the next phase.

All tasks inside a phase should be clearly defined in terms of responsibilities.

Any element resulting in blackout periods should be identified in the very beginning and the planning should be set up accordingly.

Typically such a consolidation project might consist of the following stages :

- Business requirement and needs identified:
  - Identify possible target application
  - Business Solution Assessment
  - Return On Investment (ROI) assessment
  - Business case for consolidation to OS/390
    - Define environment with OS/390
    - Proof of concept (optional)
      - Select a representative part of an application
      - Implement it and test it on OS/390
      - Analyze and present the results
  - System solution defined
  - Presentation to business and IT management
- Systems requirements and sizing:
  - Preliminary assessment of capacity and performance requirements
  - Define components of bid hardware, software, resources
  - UNIX porting assessment
- Project Definition Workshop:
  - Define success factors
  - Define configuration
  - Identify project team
  - Develop project completion proof points
- Project planning:
  - Task definition
  - Schedules
  - Resources
  - Identify dependencies
  - Identify the milestones
  - Deliverables

- Project implementation:
  - Detailed design
    - Set up the test environment
    - Transfer source code to OS/390
    - Port makefiles and shell scripts
    - Iteratively compile, test and eliminate errors
    - Create executable programs
    - Port runtime procedures
    - Load the test data
    - Test based on a defined test plan
      - Integration tests
      - System tests
      - Performance tests
    - Integrate ported application into production OS/390 environment
- Project completion:
  - Determine if proof points are met by production system
  - Analyze results to assess project success
  - Formal agreement by management to project completion
  - Identify additional new applications for consolidation (if they exist)

In a porting project, the following points should be considered with particular attention:

- Evaluation of the effort to port the code. This effort could be dependent on one or more of the following points:
  - The fact the code has already been ported to another UNIX platform
  - The availability of the people who coded the application in the source platform
  - The availability of the up-to-date application documentation.
  - The degree of compliance of the source code to standards (both in-house and public)
  - The need to convert a database from one type to another (for example, from Oracle or Sybase to IBM's DB2)
- The numerous possible involved parties such as:
  - IT operations, application development, ISV, hardware vendor
  - Architects and designers
  - System programmers
  - Network specialists
  - DBAs
  - Operation analysts
  - End users
  - Hardware planners (for capacity planning, configuration dependencies and so on)
  - Developers

  There should be a document of understanding (DOU) between the various organizations.
- Porting an application does not mean reengineering an application. The porting operation should be done and completed before introducing any new system or application functions.

- The application to be ported may involve numerous software products. During the planning phase, we suggest the following be done:
  - Identification of all the products being used by the application
  - Availability of support skills for these products
  - Availability of customer skills including a good knowledge of the whole application
- If the application has been created using a third party tool, then a prerequisite for porting the application to OS/390 may be that the third party provide OS/390 support for that product. This could apply to 4GLs, code generators, access methods, screen toolkits and so on. The support and co-operation of the third party will be essential and may not be easy to obtain.
- When a database conversion is part of the porting, it may result many new field definitions. The testing should cover these new fields.

  Decide whether to do a complete application port versus a staged approach (for example, perform the code migration with the existing database system, and then port to the new database).
- Schedule the phases of the project to coincide with business operations (for example, do not try to cutover to the new system during periods of high business activity, or at business critical times of the day).
- Be aware that there will be cultural differences between the various parties involved. UNIX culture is very different from OS/390 culture; IT people will need to have a good view of the business requirements, operations people often have different views from developers, and so on.
- Communication within the project will be similar to other types of projects, but we must reiterate the need to ensure that non-project management people are also aware of the importance of the project.
- There may well be geographical differences involved. For example, the ISVs who supplied parts of the application may be located in different parts of the same country, in different countries, and/or in different time zones. ISV's, for example, could be in a completely different part of the world.
- The architecture of the application to be ported.

  It is very important to check that the application architecture is suitable for a port to S/390. The project plan could be very dependent on whether there will be architectural changes. Some parts of the application may need to be changed to run on S/390.
- ISV

  For a software vendor, there is much more to bringing a product to the market than porting lines of code. The new market will comprise a very different customer set and may require different marketing and sales techniques. This was particularly true when the ISV offered a complete standalone solution running on many UNIX platforms, previous to OS/390 UNIX Services.
- An experienced project manager must be available from the beginning of the project.

6.1.3 Alignment of Objectives

The objectives of the project should be clearly described in a document, involving the customer business sponsorship, the customer IT sponsorship and IBM.

6.1.4 Project Sponsorship

The sponsor should be a person owning the project in terms of decisions to be taken should the project deviate from the objectives or be delayed. The IT people and business people should report to this person in the customer's organization.

Typically the sponsor will commit the resources which are affected by the project.

6.1.5 Culture - OS/390 vs UNIX

When porting an application to OS/390, it is not only the operating system that changes: the whole environment or culture within which the application will be run and managed is changed.

The new production environment may have different controls over what can be done, when and by whom. In the UNIX environment, the application may have been the only application on a particular server and minor problems may have been corrected by logging in as the root user or by rebooting the system. Such an approach would not be appropriate in a S/390 data center, where multiple business-critical applications may be running on the same server.

It is not sufficient that the code simply executes on the new platform. Systems management aspects such as capacity and performance management, availability management, service levels, testing strategy, security, backup and recovery, change control and so on must all be considered and included in the plan. Requirements must be agreed to by all parties, and processes and resources must be in place.

6.1.6 Communication

Communication between all parties (customer, IBM, ISV, contractors and so on) is absolutely vital and must continue throughout the porting process.

After the planning has been accepted by the customer and IBM, a communication plan must be defined, including two types of communication:

- Internal project communication

  All the parties involved in the porting project should have the same level of information in terms of planning, what has been done, any issues, and so on.

- External project communication (from the customer point of view)

  The customer should keep other parties who are not directly involved in the project informed of the project progress and status using the organization's normal internal communication processes.

  The reasons for, and benefits of, the migration should be communicated to all persons in the organization who have an interest in or who may be impacted by the application port.

6.1.7 Geography

During a porting process, many products and skills can be involved the application port of the customer application.

Where there are a large number of participants, multiple locations where those people are geographically located, and tasks are distributed and synchronization is difficult, then there may arise situations where time is wasted or errors occur due to a misunderstanding of tasks, objectives, status and so on.

Therefore good synchronization between all interested parties is mandatory, as well as clear definitions on what is to be done, what is required by when, and which targets are required and when.

The project manager should arrange project meetings (design reviews, status meetings and so on) in such a way that the key people are face to face, either in person or via teleconferencing.

6.1.8 Change Control

As in any project dealing with code development, the change control processes should be clearly specified.

We can identify two types of software changes:

- Changes applied to the products, in terms of version, release, fixes. These changes should follow the existing procedures for the customer's OS/390 server(s) (or good standards need to be defined, implemented and enforced for customers new to OS/390).
- Changes applied to applications, for example, new functions, correction of errors, new versions and releases.

  These changes are part of the project management, as long as the porting project is not completed.

  All changes need to be documented by the originator. The impact on the planning, and the workload to implement it, need to be presented to the customer during project status meetings. The decision to implement any change should be documented.

  Application changes should be frozen until the port is completed if at all possible. Only mandatory changes (that is, required for business reasons) should be permitted. Preferably, any application changes should be separately tested and only introduced at mutually agreed points in the migration.

The S/390 environment should have two OS/390 systems available at a minimum (either separate systems, or logical partitions of a single server: one for production, and one for testing.

The test partition should have the same system components (OS/390 version, compilers, database subsystems and so on) at the same software level as the production system. The components needed for the porting project must be installed on both systems.

The ported components should be brought first to the test platform. The system administration definitions should be consistent with the production environment. As

soon as the tests start on the test partition, any changes (system and application) should be frozen on all systems.

The testing scenarios, previously created according to the test plan, should be run and the results saved.

For each change, or set of changes, a level number should be assigned to the application components on the test system. The components of the test system (operating system, data base, application code and so on) should be backed up so that the system environment prior to the change can be restored should any production problem need to be investigated and resolved.

Any application or system change should be brought from the development platform to the test partition and tested. The test scenarios should be updated to cover the change.

For a high priority change (for example, for an end-user function does that not work properly), the failing component could be fixed by a code patch without changing the level number.

After completion, the components should be delivered to the production administrator, with the documents related to the application operations.

6.1.9 Workers Should Attend Meetings

At the beginning of the project, a certain number of meetings will be defined at various project checkpoints and reviews which will include differing participants.

The project manager must insure that all the people involved in working on the project element covered by any given meeting *must* attend the meeting. The absence of, for example, program coders could led to key project requirements not being communicated to all concerned.

The minutes and actions should be communicated by the project manager to all other interested parties (for example, the project sponsor, other management, end-users of the system, other project members and so on).

6.1.10 Staged Approach

Depending upon the complexity of the project, it may be a good approach to consider migration of application programs and data database systems separately.

This should be possible if the “from” database can be accessed from the “to” programs (that is, an OS/390 client accessing an Oracle UNIX database, with the Oracle database being migrated to OS/390 DB2 later).

6.1.11 Critical Resources

The application code porting itself needs to be carried out by programmers who have a detailed knowledge of the structure, use, and maintenance of the application (or part of the application) for which they are responsible. C programming and debugging are required, as is familiarity with the Korn shell and utilities such as make and c89. Previous porting experience is preferred and a working understanding of open systems standards is useful.

During the port, systems administration tasks such as adding new user IDs, adding new filesystems, changing customization options and so on require OS/390 systems programming skills. Experience shows that in practice a programmer may not need personal OS/390 skills, but does require access to them. Programmers need never leave the shell environment. OS/390 includes familiar UNIX utilities such as vi editor, dbx debugger, lexx, yacc, make, tar, pax, cpio, cc, and c89.

6.1.12 Migration Plan

The migration phase takes place immediately following the system tests (or performance test, if any). The new system is put into production during this phase.

Depending upon the complexity of the project, the migration phase may be very critical. This is particularly true when the customer's production environment cannot be interrupted for long periods of time. Thus the migration plan needs to be considered and developed very early in the planning process.

Some of the elements applicable to almost any project are:

- Migration conditions fully satisfied:
  - Test plan completed and results presented to the customer.
  - All the deliverables ready and delivered.
  - End users educated and IT operations trained on the new system.
  - User definition and administration done on the new system.
  - System definition and administration done on the new system.
  - Database definition and generation done on the new system.
  - Load modules generated on the new system.
  - Procedures installed on the new system.
  - Network definition (if applicable) done and tested on the new system. A system restart may be required at some stage.
  - The target system should be at the same system software level as the one one which the system testing has occurred.
  - Customer resources are committed.
- Migration steps:
  - Stop the old system.
  - Unload the old system database, if necessary.
  - Reload the new system database, if necessary
  - Start the new system.
  - Monitor the new system.

During these processes, the old system should be restarted should any problem occur during the migration.

Some decisions that affect both migration and production phases should be made very early in the project planning cycle. These include the following:

- Mode of cutover to consolidated environment: “sudden death” or parallel running.
- Application and system maintenance:
  - How should the application be maintained?
  - How is should debugging be done?
  - How is performance testing to be carried out?
  - How should the system test be performed?

- Tools and procedures:
  - What tools are needed?
  - What procedures should be changed/added?
  - What standards should be changed/added?
- Exploitation of S/390 unique capabilities:
  - Should the application be enhanced to make it sysplex-enabled?
  - Are there any other S/390 unique capabilities to be considered (remote copy, and so on)?

6.2 Education Requirements

The following education is recommended before going through a porting project, the assumption being the customer already has an OS/390 system in production.

6.2.1 Executive Management

Education should be given on OS/390 capabilities and benefits, with an emphasis on the integrated UNIX services and OS/390 benefits.

6.2.2 Project Manager

The project manager should be familiar with:

- OS/390 functional capabilities and benefits
- UNIX concepts
- Application development concepts
- Project management techniques

6.2.3 IT Staff

The IT staff should have been trained in the following areas:

- System/networking: OS/390 UNIX System Services and TCP/IP, installation and configuration
- Security administrator: UNIX security procedures, and OS/390 security architecture and OS/390 security products (RACF, for example)
- Operations: OS/390 HFS file organization, ported applications management
- Database Administrator (DBA): Database administration for the ported database

6.2.4 Application Developers (including ISV)

The people in charge of the development should be familiar with:

- UNIX operating system
- OS/390 UNIX System Services environment
- OS/390 HFS file organization
- TCP/IP functional capabilities
- File transfer utilities
- Good practices for the programming language involved (such as C, C++, COBOL, and so on)

- Programming language compiler options
- Programming language debugger options
- Shell utilities
- OS/390 target database APIs
- OS/390 target database utilities

6.2.5 Architect/Designer

- OS/390 architecture concepts and facilities
- Non-IBM database and IBM DB2 concepts and facilities
- TCP/IP concepts and facilities.

Chapter 7. OS/390 UNIX System Services Performance Considerations

In Chapter 4, “Sizing and Performance Considerations” on page 69, we discussed some general issues of performance of UNIX and OS/390 servers. In the present chapter, we discuss issues specific to OS/390 UNIX System Services in achieving better performance following server consolidation.

7.1 Performance of OS/390 UNIX System Services

In this section we discuss some system and application choices that should be understood when tuning the consolidated system.

7.1.1 OS/390 UNIX System Services Process Management

In order to set up, configure or manage OS/390 UNIX System Services application programs, you should understand the categories of work that exist in OS/390, such as:

- Started tasks (OS/390 operator START command)
- Batch JOBs (submit via the Job Entry Subsystem (JES))
- TSO/E users (logon)
- APPC/MVS transactions (CPI-C allocate)
- IMS (a started task)
- Other server or system address spaces

All the work is created from these requests and ends up in an address space. The address space holds pieces of information that describe the work, for example storage control blocks, programs to be executed and so on. The real units of work that OS/390 dispatches are the Task Control Block (TCB) or Service Request Block (SRB).

7.1.1.1 Processes

With OS/390 UNIX System Services, nothing changes except the terminology that is based on the concepts of a process and a thread.

An OS/390 UNIX System Services process maps into an OS/390 address space and an OS/390 task environment defined by a task control block (TCB) and related control blocks that exist for the process. In addition to the TCB, the OS/390 UNIX System Services kernel address space contains a number of control blocks that represent an OS/390 UNIX System Services process. There are situations where multiple processes may exist within the same OS/390 address space, and in such cases a process may be running as a job step task or subtask.

To start a new process, a process may use the fork() service as in any UNIX environment. In OS/390 Version 1, this service invokes APPC/MVS services to create a new address space for the new process, called the *child* process. There are routines and structures that are copied from the parent address space to the child address space.

Beginning with OS/390 Version 2 Release 4, OS/390 UNIX System Services changed from using APPC initiators to Workload Manager (WLM) controlled initiators. The initiator address space name is BPXAS instead of ASCHINT. BPXAS is a special initiator with special interfaces to WLM and UNIX System Services.

`Fork` and `spawn` use a WLM service to get an address space to process a request. The WLM service checks its queue to see if there is an idle address space from a free pool of BPXAS address spaces that are waiting for work. If there are no idle BPXAS address spaces, WLM determines whether a new one can be created based on system load. If the free pool is empty, an ASCRE is done using the name BPXAS as the procedure to start a new one. When a BPXAS address space is finished, the address space goes back into the free pool. After 30 minutes of being idle, the address space is terminated.

The spawn service is another way to create a new process. It looks like fork(), except that the new process is not a copy of the parent process. The child process inherits file and socket descriptors, as with the fork(). With a spawn, the child process can be created in the parent address space.

As mentioned in Chapter 8, “Security Considerations” on page 119, the _BPX_SHAREAS variable set to YES avoids the call to APPC to start the child process.

7.1.1.2 Process control

OS/390 UNIX System Services *dub* the OS/390 address space as a process. In order to do that, new information is added to the current address space:

Name	Definition
Real UID	Identifies the user who created the process
Effective UID	Determines the owner access privileges of a process
Real GID	Identifies the current connect group of the user for which the process was created
Effective GID	Determines the group access privileges of a process

The real UID and GID indicate who the user is, and the effective UID and GID are used for file access permission.

Each process has an ID and belongs to a process group. Each process group has a unique group ID. The most important attribute of a process group is that it is possible to send a signal to every process in the group just by sending the signal to the process group leader.

Note: The `kill` UNIX command must have a hyphen (-) before the PID of the process of the process group leader in order to have the signal propagated to the group.

Some process identifiers are created for job control:

Name	Definition
PID	Process ID, as unique identifier assigned to the process when created
PGID	Process group ID, as unique identifier assigned to the group. It is the same as the PID of the first process in the group.

PPID Process ID of the process (parent) who created the new process (child)

7.1.1.3 Process priorities

They are managed as follows:

- A dubbed OS/390 address has a priority based on whether it is a batch job, TSO work, started task and so on. Being dubbed does not change the priority.
- Forked or spawned processes that run in an APPC initiator address space get their priority assigned by the System Resource Manager (SRM) in the IEAICSxx and IEAIPSxx parmlib members. If the system runs in GOAL mode, the priority comes from what is defined in the Workload Manager (WLM) goals.

  In fact, child processes do not inherit their priority from the parent. They are considered members of the OMVS subsystem category. This allows tuning by account number or user ID in the appropriate SRM or WLM controls.

7.1.2 Performance Recommendations

OS/390 UNIX System Services implementation in the OS/390 environment, means the following:

- Address spaces
- Library management
- Workload management

Tuning techniques, which recognize the differences from an unconsolidated UNIX environment, must be applied in order to optimize the resources and take advantage of the S/390 architecture. Tuning can also provide improved system utilization and increased throughput.

The recommended tuning changes are:

- Storage allocation
- Sticky bit for the shell
- RACF UIDs and GIDs
- Placing frequently used modules in LPA
- APPC initiators
- Shell variables
- Propagation of TSO/E or ISPF STEPLIB data sets

7.1.3 Storage Allocation

When running OS/390 UNIX System Services in a partition or as a single OS/390 system, make sure that the system storage size is 64MB.

7.1.4 Sticky Bit for the Shell

In order to improve system performance, make sure that the shell is made resident in the Link Pack Area (LPA). The sticky bit for /bin/sh must be set on as a default option.

Frequently used shells, utility routines, C/MVS runtime library routines and compiler modules can be put in the LPA. Update the IEALPAxx SYS1.PARMLIB member in order to put the key modules into the modified link pack area (MLPA).

Note: Select the appropriate routines to put under LPA control, because too many routines can affect the amount of virtual storage available to your own OS/390 address space.

7.1.5 RACF UIDs and GIDs

Cache your UIDs and GIDs in order to have them under control of Virtual Fetch (VLF). Update the COFVLFxx SYS1.PARMLIB member to include:

```
CLASS NAME(IRRGMAP)
      EMAJ(GMAP)
CLASS NAME(IRRUMAP)
      EMAJ(UMAP)
CLASS NAME(IRRGTS)
      EMAJ(GTS)
CLASS NAME(IRRACEE)
      EMAJ(ACEE)
```

Verify that the VLF component is started from an SDSF screen or using the operator D A,L command.

7.1.6 APPC Initiators

Make sure you have enough APPC initiators defined. For that purpose, use the console with SDSF to check, because the system creates and deletes APPC initiators permanently. Define enough in the ASCHPMxx SYS1.PARMLIB member.

```
 CLASSADD CLASSNAME(OPENMVS) MIN(20) MAX(300) RESPGOAL(1)
```

A good value to start with is:

```
MAX = (MAXUIDs x 3) + 5
```

7.1.7 Shell Variables

There are two variables that can impact the performance of the shell and the utilities performance.

```
_BPX_SHAREAS=YES
_BPX_SPAWN_SCRIPT=YES
```

7.1.7.1 _BPX_SHAREAS=YES

This variable enables shared address space for the shell. If this variable is not set, the shell creates all processes in separate address spaces.

Issue the command:

```
export _BPX_SHAREAS=YES
```

or update $HOME/.profile with:

```
 _BPX_SHAREAS=YES
```

With OS/390 UNIX System Services, when the SHAREAS keyword is used, the login shell process is nested in the user's TSO address space.

7.1.7.2 _BPX_SPAWN_SCRIPT=YES

To improve performance when running shell scripts, set this variable to YES.

This variable is used by the spawn callable service to run files that are not in the correct format to be either an HFS executable or a REXX exec, such as shell scripts issued directly from the spawn callable service.

Setting this variable to YES eliminates the overhead associated with the shell-invoking spawn again after receiving ENOEXEC for an input shell script.

Set :

```
_BPX_SPAWN_SCRIPT=YES
```

in /etc/profile or $HOME/.profile.

7.1.8 Prevent Propagation of TSO/E or ISPF STEPLIB Data Sets

A statement can be added in /etc/profile to improve the shell's performance for users who enter the OMVS command from ISPF or with STEPLIB data sets allocated. This prevents excessive searching of STEPLIB data sets from the shell process to the shell command processes on exec.

7.2 Performance Improvements of OS/390 Releases

Since UNIX functionality is a relatively recent addition to OS/390, there were applications that inevitably exercised the new code in an inefficient manner. To address this, there has been an on-going program of performance tuning in the operating system itself, some aspects of which are discussed here.

7.2.1 UNIX Implementation on OS/390

Performance improvement is a common goal both for customers and IBM. The UNIX implementation in OS/390 UNIX System Services was done in steps. Each release of OS/390 brought improvements in terms of performance and functionality.

In addition to the product improvements being implemented in the six-monthly OS/390 releases, applications running under OS/390 UNIX System Services need to be tuned in order to take advantage of the OS/390 architecture.

7.2.2 Description of Improvements

Since the initial release of POSIX-compliant UNIX capabilities in MVS Version 5, there has been a continuing series of performance and functional enhancements resulting in today's OS/390 UNIX System Services.

In this section some of the performance improvements are described. These improvements cover components of the OS/390 UNIX System Services environment and products dealing with such areas as:

- Compiler
- Run times
- File system
- TCP/IP
- Kernel/RTL
- DB2

7.2.2.1 C/C++ Compiler

Significant improvements were made in the OS/390 Version 1 Release 2 C/C++ compiler. Both run-time and compile-time improvements were made.

OS/390 Version 1 Release 2 showed an improvement of 15% to 30% in C compiler run times. Improvements of 15% to 30% have been measured in the OS/390 Version 1 Release C++ compiler run time. The logic of the optimizer was improved. Large benefits can be seen for applications with multiple compile units.

In OS/390 Version 1 Release 2, the compile time was improved by 30% with precompile headers for applications using a large number of headers that do not change frequently.

Compile times were also improved by improvements in the use of HFS files by the compiler. The compiler uses HFS in-storage buffers, reducing the number of I/Os and lock contentions. There are buffers in storage for the HFS and a deferred write updates the meta-data about the file after the file is closed.

7.2.2.2 File System

- The Hierarchical File System (HFS) was improved with OS/390 Version 1 Release 1 and DFSMS/MVS Version 1 Release 3.0. The elapsed time to sequentially read a large file (400MB) was reduced by 45% and the CPU time by 90%.
- OS/390 Version 1 Release 2 with PTF UW90368 improved the performance of most file operations on HFS, including sequential write improvements on large files.

Note: Future improvements are planned for HFS file processing that will provide 60% to 80% less CPU consumption.

7.2.2.3 TCP/IP

Major improvements were made to the TCP/IP component of OS/390 Communications Server since the availability of TCP/IP for OS/390 Version 3.1.

OS/390 TCP/IP for Version 3.1: This component of the OS/390 Communication Server has seen improvements in capacity, increased efficiency and reduction of overhead. More detailed information is provided in Information APARs II08848 and II08849. These two Information APARs also provide performance tuning tips that could help to answer performance questions.

Areas of improvements and their benefits are:

- Number of Telnet sessions tested up to 10,000 concurrent users
- CPU consumption by TCP/IP reduced by 17% to 32%
- Telnet transaction CPU consumption reduced by 19% to 22%
- FTP host CPU consumption reduced by 19% to 31%
- C socket application CPU consumption reduced by 17% to 32%, with throughput improvements of up to 30%.

OS/390 TCP/IP for Version 3.2: This release includes enhanced sockets interfaces that provide significant performance improvements for critical IBM-supplied applications like the FTP server, socket applications (CICS, IMS) and user-written C-socket applications.

Areas of improvement from TCP/IP for OS/390 version 3.1 to 3.2 are:

- FTP server CPU consumption reduced by:
  - 48% to 53% for binary transfer with improved throughput
  - 40% to 61% for ASCII transfer with improved throughput
- Telnet TN3270 CPU consumption reduced by 38% for steady-state Telnet transactions
- TPC C Sockets (OS/390 send) CPU consumption reduced by 55% to 70% and throughput improved by up to 265%
- TPC C Sockets (OS/390 receive) CPU consumption reduced by 53% to 67% and throughput improved by up to 130%
- UDP C Sockets (OS/390 send) CPU consumption reduced by 50% to 66% and throughput improved by up to 18%
- UDP C Sockets (OS/390 receive) CPU consumption reduced by 59% to 71% and throughput improved by up to 128%

OS/390 TCP/IP for Version 3.3: The OS/390 Version 2 Release 4 Communications Server has further improvements over Version 1 Release 3.

TCP/IP Version 3.3 performance improvements should be up to 70%. This is due to pathlength reductions after redesign.

7.2.2.4 Kernel/RTL

The pathlength for frequently used services (Syscalls, Libcalls) has been reduced significantly in the following areas:

- Kernel enhancement in OS/390 Version 1 Releases 2 and 3 benefits the Web server environment.
- Process and thread management.
- File descriptor management.
- Socket support for SRB mode and asynchronous I/O.

The C-Run-Time Library has numerous performance improvements in OS/390 Version 1 Release 3:

- Use of stack storage
- Retrieval of environment variables
- Bulk mode support for sockets

7.2.2.5 DB2 Connectivity

The following performance and functionality enhancements are implemented with DB2 connectivity:

- Stored procedures
  - Decreased costs with less network traffic
  - May be used by local or remote applications
  - Protection of sensitive part of the application from unauthorized use
  - More programming and design choices
  - Access to non-SQL sources allowed
  - Greater flexibility for change
- Elimination of Gateways
  - Native TCP/IP for DRDA

  - Stored procedures
  - WLM and DB2
- Improved dynamic SQL
  - Call Level Interface
  - Up to 85% CPU reduction
- ASCII format table data
  - Data conversion elimination

Chapter 8. Security Considerations

Today, IT systems are the cornerstone of an enterprise. Therefore, the security aspects need to be well understood by every person in the enterprise.

A failure in security can impact system availability. Unless the system, applications and database are all protected from operational errors or mistakes and voluntary intrusions or violations, then system integrity can be compromised, resulting in exposure of sensitive data, loss of data, or interruptions to online service.

The task of security control is the responsibility of a system or security administrator. Because of the large number of possible exposures across the whole IT structure, really experienced people are required to manage the security of an IT shop.

There are two aspects of security to consider, physical and logical. *Physical security* for example, has to deal with access to the computer or I/O device room. *Logical security* is related to resources such as system files, databases, and so on.

While maintaining the security of resources is difficult in a centralized data center, that difficulty is compounded many times if security has to be managed for multiple distributed individual servers.

In a distributed environment, not only does the security of each individual server need to be managed, but there are additional sources of security exposure in the communications required between the servers.

Security is thus one of the driving forces for consolidation of multiple distributed servers into a centralized "data center" environment. Many organizations are finding it simpler to manage security by moving all distributed servers into one location.

However, unless the operating environment (which in this book is either a S/390 with OS/390, or UNIX in multiple servers) provides a sufficiently secure underlying infrastructure, then all the best efforts of the security administrators can be nullified.

The remainder of this chapter describes the differences in security mechanisms between UNIX and OS/390.

8.1 UNIX Security Mechanisms

The security mechanisms of any UNIX system have two components, physical and logical. It should be noted that logical security measures are useless unless physical security is also assured.

8.1.1 Physical Security

UNIX servers should be installed in restricted access rooms. Moreover, the machine should be physically protected. The IBM RS/6000, for example, is equipped with a keylock that prevents the chassis from being opened, and renders the reset button inoperative.

8.1.2 Logical Security

Logical security in UNIX is largely implemented by means of the User ID and Group ID. Additional security can be provided by means of access control lists (ACLs).

8.1.2.1 User ID and Group ID

In the UNIX architecture each person who has access to a system has an identification called a *user ID*. Each user ID can belong to one or more *groups*. Each group is identified by its name and *group ID*. A user ID belongs to one group at a time, and can switch to another group, provided that he is listed as a member of this group.

Each object (file or directory) in the file system has nine *permission bits*. These bits are actually a subset of the mode word associated with the file. The *mode word* gives a file different attributes.

The nine permission bits are divided into three sets of three bits each:

- The first set of three bits shows the owner's permission.
- The next set of three bits shows the permission of the other users in the group to which the file belongs.
- The last set of three bits shows the permission of anyone else with access to the file.

The three bits in each set indicate, respectively, read, write, and execute permission of the file. *Execute* permission of a directory lets you search a directory for a specified file. *Read* permission of a directory lets you list the file and subdirectories it contains. *Write* permission of a directory allows you to create an object in that directory.

There are many subtleties associated with permission bits. Prior to porting UNIX applications to OS/390 UNIX System Services, make sure that permission bits and mode bits have the same semantics both under the UNIX from which you are porting and under OS/390 UNIX System Services. Check both the UNIX command reference manual (a full discussion of permission bits and mode bits is often given in the `ls` command manual page) and the `chmod` or `ls` commands in *OS/390 OpenEdition MVS Command Reference*, SC28-1892.

8.1.2.2 Access Control Lists (ACLs)

ACLs are a facility offered by some versions of UNIX, including IBM's AIX. The need for the functionality they provide is as follows.

Most of the time, a system administrator team can achieve good security by simply partitioning the user set into several groups. Most users need only one group. Sometimes, however, some users need to be members of several groups but remember that they can belong to only one group at a time). Of course, they can switch between these groups with the UNIX `newgrp` command, but experience shows that this is not a practical process.

For example, suppose a user belonging to groups A and B starts in group A at login, works for a while, then starts working on tasks relative to group B but suppose he forgets to switch to group B with the `newgrp` command. The user creates a file intended for group B, but it ends up belonging to the wrong group.

Thus, other members of group B cannot access it. This kind of mishap can lower the productivity of a whole organization.

That is where Access Control Lists (ACLs) come in. An ACL is a *list of permissions* attached to an object (file or directory). These permissions permit or deny access to the object. They can specify user IDs or group IDs, and allow or deny them separately read, write or execute rights. The degree of control and flexibility offered by ACL allows system administrators to answer troublesome user requirements where some users fulfill different roles and participate into different workgroups. Using ACLs allows these users to work with files without having to switch to different groups.

An object can have at most one ACL attached to it. If there is no ACL, its permission bits are used to determine its access attributes.

Conversely, the same ACL can be duplicated and be applied to several files, to which it will grant identical permissions.

See the description of `aclget`, `acledit` and `aclput` in *OS/390 Security Server (RACF) Command Language Reference*, SC28-1919 for more information. See 8.2.3, “RACF” on page 122 for a discussion of how to emulate the functionality of an AIX ACL under OS/390 UNIX System Services.

8.1.3 Auditing

Several versions of UNIX, including AIX, have full auditing. The system administrator can specify which objects (files or directories) are to be audited, and what kind of access should be audited. Audit log files are produced during the operation of the system. They appear as text files to the operator.

Ideally, these logs should be reviewed periodically. Unfortunately, there is no standard way of filtering and querying audit log files under UNIX. For this task, most system operators rely on locally-written scripts in AWK, Perl or other text-processing languages.

8.2 OS/390 Security Mechanisms

OS/390 has very good security mechanisms. OS/390 UNIX System Services rely extensively on them, especially on RACF, to implement its security mechanisms.

8.2.1 Physical Security

Most OS/390 systems today reside in traditional data center environments, where for many years physical security has been a prime requirement.

Physical security of equipment in the data center is covered by controlling access to the room where the computer and I/O devices are installed. Generally these rooms have restricted access to the few people who have specific tasks like maintenance, installation or modification which require physical access to the facility.

Frequently, large customer data centers are operated in “lights out” mode, where the machine room(s) are only entered for reasons of maintenance, physical re-configuration and repair.

In 1990, IBM introduced the Enterprise System Connection (ESCON) architecture and products which use fiber optic cables for I/O connection.

The ESCON architecture allows processors and I/O devices to be separated by up to 43 kilometers, depending on the device type and capabilities. This attribute can be used for physically separating the processor room and the I/O rooms, with a consequent increase in physical security (as well as enabling new, and different, approaches to disaster recovery; see 9.3.1, “OS/390 and Disaster Recovery” on page 127).

8.2.2 Logical Security

Logical security is related to operating system resources such as system files, databases, and the operating system itself.

In the OS/390 architecture, each person who can access a system has an identification called a *user ID*. Then, for each user, there is an assigned profile (for example for a system programmer dealing with the system definition, or for an operator dealing with daily operations.

Different groups are defined in the organization and different types of authorizations are assigned to each group. The user is part of one group and gets the authorizations the group has.

Under OS/390 UNIX System Services, as with all other UNIX systems, each file has a user ID, a group ID and a set of permission bits, as explained in 8.1.2.1, “User ID and Group ID” on page 120.

For access control, the file system relies on RACF. RACF is also used for auditing, as in 8.2.3, “RACF”

8.2.3 RACF

Under OS/390 UNIX System Services, every access is controlled by RACF. In order to modify his RACF profile, the user goes into TSO and invokes RACF.

8.2.3.1 Porting ACL-like Security

It is recommended that each user ID has it own HSF (or directory tree) mounted as its home directory. RACF has separate authorization profiles for each HSF.

As we have seen previously, AIX ACLs allow different files (even in the same directory) to have different authorizations, while RACF has one profile per HSF. Thus, you do not have the possibility of specifying different RACF authorization for files in the same HSF, although you can still specify different permission bits.

If you have to migrate an application from AIX to OS/390 UNIX System Services, you might encounter a few files that are protected by ACLs. In this case, under OS/390 UNIX System Services, you should put these files into their own HFS (which translate into their own directory from an application standpoint), and have as many HFSs as there are different ACLs.

8.2.3.2 RACF and Auditing

OS/390 UNIX System Services has a hierarchical file system (HFS) that can be represented as a tree of directories, starting at a root node. Each directory can contain files or other directories. The HFS structure of UNIX is familiar to any user of OS/2, DOS or Windows.

Under OS/390 UNIX System Services, auditing is done with RACF. RACF can audit file read or write operations, as well as directory search operations. Every object (file or directory) can be audited separately. Every access can be logged, whether it is authorized or not.

The security rules for the files and directories are stored within the file system itself in the File Security Packet (FSP), which includes the permission bits of the object being accessed. When a security decision is needed, the file system calls RACF and supplies the FSP. RACF then makes the decision, does the auditing, and returns control to the file system.

RACF generates audit log files that can be inspected and queried. Of course, system programmers can generate text files from the audit logs with a utility such as RACF SMF Data Unload Facility, and write their own tools to work on these text files. However, most of them rely on utilities supplied with RACF, such as:

- Standard RACF commands such as LIST and SEARCH
- IRRUT100, the RACF Cross-Reference Utility
- IRRDBU00, the RACF Database Unload Utility
- RACFRW, the RACF Report Writer
- DSMON, the RACF Data Security Monitor

You can find more information on these utilities in the redbook *OS/390 Security Server Audit Tool and Report Application*, SG24-4820.

Chapter 9. Operational Considerations

Server operations are important because they directly affect the quality of service delivered to the end users and other divisions in the enterprise.

Operations can be viewed as a set of habits and recipes that make use of the available system facilities. In order to migrate smoothly from the UNIX to the OS/390 UNIX System Services environment, it is important to be aware of all the operational procedures of the existing site. The cultural gap between UNIX and OS/390 is so wide that many people are amazed when they discover the habits and methods of the other group.

This section is intended as a side-by-side comparison of the two cultures when dealing with the same problems. Readers familiar with one of the cultures can thus discover how these problems are tackled in the other world, and estimate how proper operations will be possible after the transition.

Operational management includes different disciplines that a customer must consider in a computing environment: Operational management may vary, depending on the following factors:

- Customer size
- Customer IT geography
- Principal operating system (OS/390, UNIX, VM or VSE)

9.1 Operational Differences

In this section we consider the main characteristics that differentiate the OS/390 and UNIX environments.

The aspects of the operational environment that we discuss are:

- System administration
- Security
- Contingency and disaster
- Recovery
- Automation
- Batch
- System performance
- Capacity planning
- Storage management
- Configuration management
- Problem management
- Change management
- General considerations
- Additional considerations

Each of these is discussed and compared.

9.2 System Administration

The successful operation of any server environment depends very strongly on the extent and quality of the management of it. Areas that must be strongly monitored and control include: performance, capacity, storage usage, database administration and network management.

There major differences between OS/390 and UNIX in both the approaches taken to systems management, and in the tools available.

9.2.1 OS/390 Environment

Because the OS/390 environment covers different types of activities involving people it requires a clear organizational structure. Activities are assigned to specific people with defined responsibilities, keeping in mind that communications must exist between all parties.

Depending on the installation size, there could be many people involved in IT management, each with dedicated responsibilities, such as:

- System and Security administrator
- Storage administrator
- Database administrator
- Network administrator

9.2.1.1 System and Security Administrator

Normally there is a dedicated person called the system administrator who is responsible for defining the people who will have access to the production system.

User access to, and use of, the system is controlled by the authority given to the user's ID, groupings of users with common requirements, and the interaction between user groups.

9.2.1.2 Storage Administrator

The Storage Administrator's role consists of planning, defining and checking the use of both disk and tape storage resources.

9.2.1.3 Database Administrator

The Database Administrator task consists of planning, defining and checking the resources in terms of space and availability of databases.

9.2.1.4 Network Administrator

The Network Administrator role involves planning, defining and checking network resources (lines, routers, and son on). It includes the provision of the network connections between the users and the server(s), and maintaining service in the event of a communications failure.

9.2.2 UNIX Environment

9.2.2.1 System, Security, Storage and Network Administrator

In the UNIX environment, there was originally only one privileged user ID, *root*. The root user can access all the resources in the machine without restrictions. He/she plays the role of system and security administrator. But on small systems, he/she also fills the roles of storage and network administrator.

On large systems, these different roles quickly become too heavy a burden. Thus, different classes of administrative privilege have been added to UNIX for the purpose of delegating some tasks to other people who do not necessarily need to have root privileges. But they are not the default configuration, they have to be configured. The default configuration is to reserve all these roles (system, storage and network administrative tasks) for the root user (whose password is generally known only by a handful of people).

Because for most UNIX setups, the system, storage and network administrative tasks require login as root, there is no strict delineation of security privileges between these tasks. This lack can be compensated for by close coordination between members of the administrative team.

9.2.2.2 Database Administration

The database administrator has the same role in UNIX as in the OS/390 environment. He generally does not need to have root access, since most database subsystems have their own administration profiles.

9.3 Disaster and Contingency

Customers should have a plan for continued provision of IT services in the rare event of a catastrophic situation, for example, where the IT center is damaged by fire, flooding or any other kind of problem that could destroy the enterprise data and access to applications.

Most S/390 sites have such plans in place and regularly exercise their proper operation. However. many small sites (particularly those owned and operated by end-user departments) do not have such a recovery plan, or if they have one, have not exercised it.

Such sites are at risk if a disaster occurs, and many experts in this field estimate that the majority of such businesses would fail immediately, or within a few years, as a direct consequence of the disaster.

This is another driving force behind the trend for consolidation of multiple distributed servers into a centralized IT-managed location where proper disaster recovery plans can be more readily implemented.

9.3.1 OS/390 and Disaster Recovery

Large-system customers have been sensitive to this problem for many years. Disaster recovery planning and implementation is just another task that an IT shop must perform as part of its responsibilities.

The OS/390 environment offers various possibilities for disaster recovery, including:

- Backup sites
- Parallel Sysplex

9.3.1.1 Backup Sites

This solution needs additional resources in terms of processor and I/O devices in order to quickly restore the processing environment, partially or totally.

Because of this, it can be a costly solution, unless those resources are operational. Usually, customers select the most critical applications to be run on a backup site. Others use old generations of processors to accomplish this task but often find maintenance on older machines to be expensive.

Regular testing is required in order to be sure that the backup processor can handle the recovery with minimal impact on crucial applications.

9.3.1.2 Parallel Sysplex

IBM introduced the Parallel Sysplex in 1994 as the solution for S/390 clustering for very high availability and virtually unlimited scalable performance.

In a Parallel Sysplex, up to 32 processors (each of which may be a 10-way SMP) are connected using a Coupling Facility and Coupling Links to provide a shared-data, single-system image no matter how many processors make up the complex.

Since all processors share access to the data base(s), multiple copies of the same application can be run on multiple processors within the Parallel Sysplex.

Should additional capacity be required, then additional copies of the application can be started.

Should any component required to run an application fail (for example a server, the operating system, instance of the application code, and so on), then users of the application see no interruption in service since other copies of the application continue to run unchanged, and the workload is dynamically redistributed between the remaining copies, transparent to both the users and the application code.

Recently this concept has been extended to allow the processors to be at different sites, separated by distances of up to 20 kilometers (see Figure 38 on page 129).

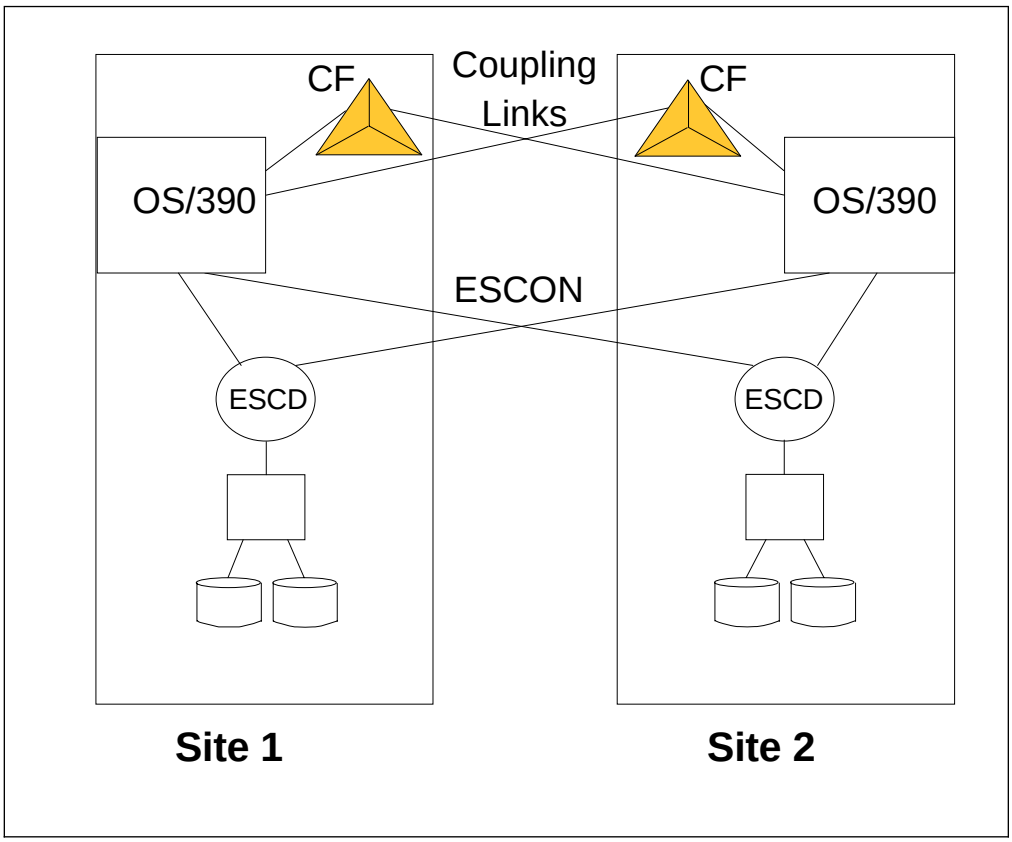


Figure 38. Extended Parallel Sysplex for Disaster Recovery

In this configuration, the Parallel Sysplex cluster is geographically extended (up to 20m) with:

- Workload balancing within Parallel Sysplex
- Tape and DASD connected by remote ESCON channels to both sites
- Dual application transaction logs between sites

This is conceptually similar to having a remote backup site with copies of the application at both sites. In this case, however, both sites have access to the shared data in one site over ESCON links.

Thus, if any server fails, the surviving server has full access to the database(s). The application continues execution, unaware that access to the main site has been lost.

The principle limitations are the cost of providing the intersite fiber optic connections, and the performance impact of the remote data and coupling facility access.

If two copies of the data base are maintained (one per site) using IBM's Peer-to-Peer Remote Copy (PPRC) or EXtended Remote Copy (XRC), then even complete loss of one site is possible with minimal interruption to service.

PPRC maintains a synchronized copy of files at each site, so that only incomplete transactions are lost, and these will have to be reentered.

XRC allows longer distances than PPRC, but provides better remote copy performance by making unsynchronized copies. Restart using database logs is necessary in this case.

In order to implement this solution, the following are required:

- Two separate processors with Parallel Sysplex enabled
- Two separate Sysplex timers
- Two separate Coupling Facilities
- IBM Remote Copy has been implemented

The connections between both sites use optical fibers for:

- Channels (ESCON architecture)
- Coupling links
- Timer links

9.3.2 UNIX and Disaster Recovery

Historically, the first UNIX application was a text-entry program. In spite of this business-oriented inception, UNIX has for a long time been perceived as a laboratory and research system. The emergence of UNIX as a general purpose business system is quite recent (within the past decade) compared to its 25-year history as a research and education tool.

Until the advent of UNIX-based business solutions, there was little incentive to consider disaster recovery requirements. That is the prime reason why failure recovery was not of concern in the UNIX world for a long time. Only recently have disaster recovery solutions appeared for UNIX machines. Since UNIX evolved on inexpensive systems, the philosophy was that reliability could be achieved through redundancy. Adding more boxes was a relatively cheap way of guaranteeing continuous operations.

9.3.2.1 Outsourcing

Several IT vendors offer UNIX recovery solutions based on a duplication of the customer's data on a separate, geographically distant machine. IBM, for example, offers its Business Recovery Service for RS/6000 SP (Scalable POWERparallel) systems.

9.3.2.2 Backup Sites

A backup site solution generally involves setting up a WAN telecommunication line and using it to duplicate data (preferably in real time) between the two sites. Should the main site fail, the backup site would take over with minimal delay.

This generally means that the customer's database has to be migrated to a backup site and updated in real time. Update transactions should be sent separately to both sites.

IBM's High Availability Geographic Cluster (HAGEO) for AIX is a good example of this kind of solution. HAGEO provides a flexible, reliable solution for automatic real-time disaster recovery. HAGEO extends the loosely-coupled clustering technology of RS/6000 SP to encompass two physically separate data centers. Each data center maintains an updated copy of essential data and runs key applications, ensuring that mission critical computing resources remain continuously available at a geographically separate site, should a planned (maintenance) or unplanned (disaster) event disable an entire site.

An HAGEO cluster consists of two geographically separated sites, each capable of supporting up to four high-availability cluster system nodes. There are three modes of disaster protection and one mode of recovery: remote hot backup, remote mutual takeover, concurrent access, and remote system recovery.

Remote hot backup
A remote geographic site and system are designated as the hot backup site and system. The backup system includes hardware, system and application software, and application data and files. In the event of a failure, the failed system's application workload automatically transfers to the remote hot backup system.

Remote mutual takeover
Geographically separated sites and systems can be designated as hot backups for each other. Mutual takeover allows each to operate as an independent system or as part of a distributed system. Should either experience a failure, the other acts as a hot backup and automatically takes over the designated application workload of the failed system.

Concurrent access
Nodes at geographically separated sites can have simultaneous access to the concurrent volume group and the same disk resources. If a node fails, the availability of the resource is not affected.

Remote system recovery
After a failed information system site has been restored to operation, HAGEO can resynchronize and reintegrate the failed system with the remote hot backup system. HAGEO updates the remote failed system with a current up-to-date mirror of application data and files processed by the backup system after the failed system ceased operations. Upon completing restoration of an up-to-date data and file mirror, the HAGEO cluster resumes synchronized system operations, including the mirroring of real time data and files between the system sites.

For more information on HAGEO, see:

`http://www.rs6000.ibm.com/software/Apps/geocluster.html`

9.4 OS/390 and Recovery

IBM has continually improved the recovery capabilities of the S/390 Enterprise Servers, adding new features or techniques.

Recovery deals with reliability and availability. Two aspects of the recovery have to be considered:

- Hardware recovery
- Software recovery

9.4.1 Hardware Recovery on S/390

This has been steadily improving for years, adding more value to each processor generation. These improvements have been made in order to make the errors transparent to the user.

Today's functions include:

- ECC on storage

  ECC codes are used to protect data in memory. These codes can detect all multiple-bit errors, and can correct in-flight, all one- and two-bit errors and most multi-bit errors.

  On the latest S/390 G3 and G4 servers, the machine can dynamically replace a chip that exhibits a high number of storage errors with a spare one.

- Parity checking on internal circuits

  Since the 360 series in 1964, parity checking has been implemented in order to avoid corrupted data being sent through the processor data flow. Upon detection, this kind of error starts the instruction recovery process in order to check if the error is temporary or permanent.

- Double operation execution

  On the latest S/390 G4 servers, a process has been implemented whereby each instruction is twice executed in parallel, then the results are compared and retried if a mismatch occurs.

- Dynamic replacement of a failing instruction processor on S/390

  On the S/390 G3 Servers and later, a failed processor unit (PU) can be replaced by a spare PU. On the G3 servers, the PU replacement needs a Power On Reset (POR), but on the recently announced G4 servers, the replacement is dynamic.

- Dynamic replacement of a failing I/O processor on S/390

  The S/390 G3 and G4 servers have the capability of dynamically replacing the System Assist Processor (SAP), which manages the storage and I/O part of the server.

- Redundant power supplies (N+1)

  S/390 servers have an N+1 design in the system power supplies. If a power supply fails, there is still sufficient power available to continue processing without interruption. The failing component can be replaced and restored to use without stopping the operation of the server.

Note: The S/390 G3 and G4 servers have spare PUs in order to provide dynamic replacement of a processor or SAP when needed.

9.4.2 Software Recovery on OS/390

In addition to providing support to assist the hardware in retrying and recovering from failing hardware components, OS/390 provides an extensive set of software recovery aids.

For example, OS/390 has standard completion and wait state codes, system messages to both the operator and application, and very comprehensive hardcopy and online documentation of these.

Note: Most installations use IBM or other vendor-supplied automation products to capture and respond to the majority of software error conditions, leaving the operator free to handle only exception conditions.

OS/390 software design also incorporates a unique mechanism known as an Functional Recovery Routine (FRR). Each major OS/390 component has a FRR, which is a routine that is given control should the component fail for any reason.

It is the responsibility of the FRR to analyze the cause of the software failure, to perform cleanup of the environment, and to retry and recover if possible.

Should the FRR be unsuccessful, the component is terminated and control is passed to the FRR of the component that invoked the failing software.

In this way, OS/390 can trap and, in the majority of situations, recover from its own (or subsystem) failures so that, even though an individual application may be interrupted, the remainder of the system continues to provide service to the endusers.

9.5 UNIX and Recovery

Here again, mentality and historical differences are perceptible in the way the UNIX world approaches the recovery problem, compared to the mainframe world. Since its inception, UNIX has been designed to be able to run on small, inexpensive machines. Thus, instead of trying to make a single processor failure-proof, the UNIX approach has been to duplicate the processors. However, in the last few years, fault-tolerant UNIX machines have appeared.

9.5.1 Hardware Recovery on UNIX

Of course, OS/390 UNIX System Services benefit from all the hardware recovery techniques on S/390 servers discussed in the previous section. Other, non-S/390 UNIX machines also use some hardware recovery mechanisms.

- ECC on storage

  Most recent motherboards use ECC to detect and correct memory errors, which is now implemented at the memory board level. The ECC provides detection of double-bit errors and correction of single-bit errors.

- Parity checking on internal circuit

  The POWER processor imbeds error detection circuitry that prevents incorrect data from being processed.

- Disk Drive Predictive Failure Analysis (PFA)

  Some IBM disk drives, such as the ones used in the RS/6000 Model F50, incorporate the PFA technology. The built-in PFA circuitry continuously monitors drive performance, and can determine an early tendency toward disk drive failure long before it actually happens, based on disk drive conditions and operating parameters. When this condition is detected, an error status byte is returned to the AIX device driver, which results in an error log entry in the AIX error log facility. Automatic Error Log Analysis, if enabled, sends a warning message to the system administrator. Preventive measures can then be taken to preserve the disk data and perform a repair action before a serious impact to customer operation occurs.

- Other hardware improvements

  Some UNIX machines now include fault-tolerant hardware. For example, IBM's RS/6000 Model F50 incorporates seven new Reliability, Availability and Serviceability (RAS) design features:

  – Method and system for reboot recovery
  – Environmental and power error handling extension and analysis
  – Automatic processor surveillance
  – Machine check handling for fault isolation in a computer system
  – Method and system for check stop error handling
  – Error collection coordination for software-readable and non-software-readable fault isolation registers in a computer system
  – A method and system for fault isolation for PCI bus errors

- Processor redundancy

  Some of the current UNIX systems can run on either SMP machines or loosely-coupled parallel systems and use some form of take-over redundancy.

  For example, the High Availability Cluster Multi-Processing (HACMP) facility for AIX (Version 4.2) is a control application that can link up to eight RS/6000 servers or SP nodes into highly available clusters. Clustering servers or nodes enables parallel access to their data, which can help provide the redundancy and fault resilience required for business critical applications. Uniprocessors, symmetric multiprocessors (SMPs), and RS/6000 SP nodes can all participate in highly available clusters.

  HACMP automatically detects system failures and recovers users, applications and data on backup systems, minimizing downtime to minutes or seconds. In addition, using HACMP for AIX minimizes planned outages, since users, applications and data can be moved to backup systems during scheduled system maintenance.

  With the addition of the new Dynamic Reconfiguration facility, customers will no longer have to stop their cluster to add or remove processors, adapters, or networks. Using administrative interfaces, these activities can now be performed on an active HACMP cluster. Dynamic Reconfiguration can also be used to change the failover behavior of the cluster without disrupting current operations. The new Cluster Single Point of Control (CSPOC) facility lets customers perform common cluster administration tasks from either node in a two-node cluster, regardless of the current ownership of cluster resources. These enhancements to HACMP allow cluster configurations to be modified without cluster stops and restarts, helping the customer approach non-stop cluster operation.

  HACMP clusters can be configured in several modes for different types of processing requirements. *Concurrent access* mode suits environments where all of the processors must work on the same workload and share the same data at the same time. In *mutual takeover* mode, the processors share the workload and back each other up. Idle standby allows one node to back up any of the other nodes in the cluster.

  When a failure occurs, or when components are restored to operation, HACMP executes test and recovery scripts tailored by the system administrator to specify the actions to be taken. HACMP software detects and recovers from failures of disks, disk adapters, networks, network adapters, and processors.

HACMP relies on the application, however, to make any failure recovery or fallover transparent to external users and client machines. If a node fails, nominal recovery time is approximately 30 to 300 seconds. Actual recovery time is a function of the system configuration, the application configuration, the size of the user's databases, and the user's recovery script (if any).

For more information on HACMP, see:

`http://www.rs6000.ibm.com/software/Apps/hacmp/`

9.5.2 Software Recovery on UNIX

There are two separate kinds of error reporting in the UNIX environment: application error reporting and system error reporting.

9.5.2.1 Application Errors

All UNIX systems have a list of standard error numbers, listed in the header file `errno.h`. When an application or a system call reports one of these errors, it usually mentions the error number (or errno). The operator knows what kind of error happened, but not necessarily what resources are involved. For instance, a badly-written application can report that it cannot open a file by simply returning the errno value of the last failed system call. In this case, the name of the file that could not be opened does not show up, and only a programmer familiar with the application (or running a trace tool) could point to the error.

Thus, application programmers have to take great care with the kind of information they return in case of error. It is up to them to supply details of the problems their code encounters. During the porting of an application from UNIX to OS/390 UNIX System Services, the programmers ought to make sure that the level of information supplied in case of errors is sufficient to allow the system operators to correct it.

Commercial database management systems (DBMS) generally provide adequate error reporting. The database administrator should work in coordination with the operators to make the best use of the error report facilities of their DBMS.

9.5.2.2 System Errors

System errors are the kind of errors encountered by non-application processes, such as daemons. (*Daemons* are, in UNIX jargon, unattended tasks running in the background.) Daemons trace their errors to the console, and often to a specific file, depending on the daemon. It might be useful to write a script for watching these error files.

Some daemons use the syslog facility. This is done by adding definitions in the /etc/syslog.conf file, where one can specify the type of errors traced and the error log file where they are traced. A good error consolidation policy should thus watch all the error logs declared in /etc/syslog.conf.

Some UNIX systems also trace abends or transient errors. For example, AIX maintains an error log that can be queried with the `errpt` (error report) command. This command can be used to display errors according to specific criteria (classes, devices, dates, and so on). The errors are stored in binary form in an error log (by default, /var/adm/ras/errlog.) An error log analysis can be conducted with the `diag -e` command.

9.5.2.3 Differencies between UNIX and OS/390 Error Management

Philosophical differences between UNIX and OS/390 show up again in their respective approaches to error management. During the 1960s and early 1970s, the OS/360 approach was to log all program errors to the operator console. Then, as systems grew, the number of error messages overloaded the operators (whose job was not to maintain the applications anyway). So the console logging approach (based on the Write To Operator assembler macro) gave way to OS/390's automated error handling methods.

UNIX systems, in turn, are now trusted with an increasing number of simultaneous applications. This is especially true in SMPs or parallel systems, where the operator console can receive messages from hundreds of applications running on dozens of processor nodes. So UNIX developers are now abandoning the classical console error logging in favor of automated error handling packages such as NetView.

9.6 Automation

Using automation is important for helping the operational teams. Today there are automation products in the marketplace that help customers in managing the production tasks.

The purpose of automation is to:

- Help the operator manage repetitive tasks
- Avoid wrong decisions during sensitive situations
- Improve system availability (24 hours a day and 7 days a week is the objective)

The production staff is mainly concerned with automation, because it also impacts system availability.

9.6.1 OS/390 and Automation

In the OS/390 environment, automated operations have been implemented for many years in order to simplify systems management. There are many different products in the marketplace today providing such automation.

IBM offers:

- Job Schedulers, such as Operations Planning and Control (OPC/ESA)
- System Automation Manager for OS/390, which contains three (formerly separate) products:
  - Automated Operation Control (AOC/MVS) which automates console operation.
  - Target System Control Facility (TSCF) which provides the ability to operate multiple systems from a remote location.
  - ESCON Manager (ESCM) which automates configuration management for ESCON (fiber optic) channels, switches and attached devices.
- NetView, for automation of both local and network operations.

It should be noted that all such products, offered either by IBM or by other vendors, are tightly integrated with OS/390, and are usually interoperable with little overhead. The level of integration is generally lower with UNIX automation tools.

9.6.2 UNIX and Automation

UNIX automation and system administration tools are now offered by many vendors. The tools are very diversified, but they mainly fall into these categories:

- Job schedulers
- Distributed system management (like IBM's Distributed System Management Interface Tool (DSMIT)
- Network management tools like Tivoli TME10 (now integrating NetView).

A list of UNIX automation software packages can be found in the Buyer's Guide searchable database maintained by the UNIX Review magazine, at the following URL:

`http://www.UNIXreview.com/db_area/search.shtml`

9.6.3 Tivoli

Today, IBM laboratories and Tivoli people are working on Tivoli and IBM products in order to provide a global and integrated solution to customers' needs.

Tivoli has products in the TME 10 series which offer various possibilities in terms of system management and system automation.

TME 10 GEM: TME 10 Global Enterprise Manager (GEM) was recently announced, and it enables management and control of multiple applications in a business context.

TME 10 GEM has the following features:

- Application Policy Manager
- TME 10 GEM Management Integrations Services

The Application Policy Manager is a Java-based application for managing the business systems. This component runs on any platform that supports Java 1.02.

TME 10 GEM Management Integrations Services provides bidirectional communication between the operations management of the S/390 and distributed environments. TME 10 GEM Management Integrations Services supports the following program levels:

- OS/390 Release 3 and
  - OS/390 R3 RMF
  - OS/390 R3 Security Server
- Information/Management Version 6 Release 3 for MVS/ESA
- TME 10 NetView for OS/390
- NetView Performance Monitor Version 2 Release 3 for MVS
- ADSTAR Distributed Storage Manager for MVS Version 2.1
- IBM OPC/ESA Version 1 Release 3.1
- TME 10 Enterprise Console 3.1
- TME 10 User Administration
- TME 10 Security Management

- TME 10 ADSM
- TME 10 Performance Management
- TME 10 Distributed Monitoring

Note: For TME 10 GEM Management Integrated Services, with the exception of the security services and the performance service functions, the minimum service levels supported are:

- MVS/ESA Version 5 Release 2.2 and OpenEdition MVS
- IBM TCP/IP Version 3 Release 2 for OS/390 or any functionally equivalent product

TME 10 OPC: This Tivoli product offers a solution to customers for managing the workload on distributed systems. TME 10 OPC offers a single point of control and has a graphical user interface. Migration from OPC/A and OPC/ESA are possible through a migration aid.

9.7 Batch

Batch applications are generally used to satisfy large-volume processing.

Another characteristic of batch workloads is the number of I/Os processed during a job execution. This applies to commercial batch-type workloads. For this reason, the elapsed time of a job is dependent on the number of I/Os, the I/O configuration, and the I/O response time.

9.7.1 OS/390 and Batch

Historically, batch applications were the first workloads run on OS/390 (then known as OS/360). In the beginning, customers submitted batch jobs to run their applications. Later the interactive process took place with online applications running during the day. So the operations for a standard IT shop were:

- Online transaction processing during the day (read and update DB)
- Batch during the night (processing and refresh DB)

Note: This does not prevent any batch activity during the day concurrently with the online processing.

As mentioned before, there are many factors that impact the batch processing, most of them related to the I/O. Since the mid-1980s, many large customers encountered limitations in the overnight batch window processing. That meant the batch jobs executed to process the databases during the night did not finish before the start of daytime online processing.

In order to help customers solve that particular problem, IBM implemented techniques in OS/390 and associated subsystems to store data in buffers located in memory (central storage and expanded storage). These buffers are located in dataspaces or hiperspaces. Corresponding changes were added in the architecture.

9.7.2 UNIX and Batch

The out-of-the-box, standard UNIX has very little support for batch application, since its natural environment is the interactive process.

Over the years, many batch support tools have appeared on UNIX. They include:

- Batch submission packages, mainly used for creating and executing job chains
- Load balancing tools, used to balance the batch workload over a cluster of UNIX machines

Under UNIX, job completion reports have traditionally been sent through E-mail to their owner. This mechanism is clearly insufficient, especially in the case of distributed tasks belonging to the same batch chain. Each vendor has developed its own job reporting method, which results in little or no interoperability between batch management packages.

For example, IBM Job Scheduler for AIX allows the administrator to define a workload (chains of jobs), set rules for its execution (machines, time of the day, maximum run time, valid return codes, and so on. It allows administrators to define dependencies between jobs and files, and to set up corrective actions that are automatically triggered in the case of errors.

IBM LoadLeveler is a distributed network-wide job management program for dynamically scheduling work on IBM and non-IBM processors. From a single point of control, you can dispatch jobs to be run on a variety of UNIX machines. LoadLeveler allows you to balance workload of interactive sessions, as well as serial and parallel batch jobs. LoadLeveler can accept requests from Job Scheduler and OPC/ESA, a commercial job scheduling solution for the mainframe.

9.8 Performance

There should be a service agreement between the end user and the IT shop. This agreement is very important, because it defines the services and provides a standard of measurement of performance that the IT department offers to the enterprise. For example, the user might feel subjectively that response time is poor, but only regular monitoring against a service target will give an objective evaluation.

Systems performance administrators require tools to track and identify system problems.

Some indicators that should be tracked include:

- Response time
- Application availability
- Data ready on time

9.8.1 OS/390 and System Performance

Performance measurement requires both a skilled person and tools to:

- Define the objectives
- Measure the performance
- Detect deviations
- React to problems

There are different possible origins of performance deviation in a S/390 environment:

- A bottleneck in the processor itself
- A high processor load
- Problems on I/O devices and channels
- Problems in the network

To detect such problems, you can use:

- System indicators
- Reports (online or paper) such as RMF or others (CMF, OMEGAMON and so on)

You can define the policies at the OS/390 level in two ways:

- In the “old” way, with *compatibility* mode using the SYS1.PARMLIB members (IEAICSxx, IEAIPSxx and IEAOPTxx).
- In the “new” way or “goal” mode. This has the advantage of defining the objective in a way that the user can easily understand, such as “80 percent of my transactions in less than 1 second,” for example.

9.8.2 UNIX and System Performance

UNIX comes with an array of system tools that can be used to identify the most common performance problems. These tools can measure the various I/O activity, the processor load and the ratio of user versus system code being executed.

Generally, for a given task (which can comprise many processes), performance problems can arise because of:

- High processor load
- High I/O device load
- High paging rate
- Problems in the network (one of which may be a saturated network)

Standard UNIX tools can be used to identify these problems. Vendors of parallel or SMP machines generally deliver tools that can consolidate the performance results of the different nodes or processors onto a single display, facilitating the system administration. Applications or DBMSs can also use load-leveling facilities supplied by a vendor and make optimal use of a cluster of machines, thus minimizing the response time.

9.9 Capacity Planning

This part of operations allows system capacity to be defined based on the business forecast in order to plan future upgrades or changes in the IT configuration. Some performance problems can be avoided when capacity planning properly anticipates business growth.

A capacity planner should discuss this subject with the business managers (to gather growth data) and with the system operators (to gather system performance indicators).

The IT department should always be able to satisfy the capacity requirements of the peak periods that arise during a normal business year.

9.9.1 OS/390 and Capacity Planning

This task must be under the control of experienced people because the system behavior must be understood in order to identify where to add more resources to accommodate the growth.

There are tools in the marketplace to do that job. IBM offers tools for capacity planning in the OS/390 environment (such as CP/90 and CP2000) based on information gathered during operational periods (SMF).

There are also internal tools that IBM representatives and system engineers can use with customer data to aid in capacity planning, such as:

- Quicksizer, for sizing coupling facilities and processors in a Parallel Sysplex
- OS/390 USSIZER, for sizing the processor required for applications being moved from UNIX to OS/390

Use of these tools is restricted, because of the skills required to interpret the output and because of the rapidly changing technology in both UNIX and S/390 environments.

IBM personnel can obtain access to these tools as follows:

- Quicksizer - obtainable from MKTTOOLS as SPSSZR PACKAGE
- USSIZER - contact your local perfromance specialist

9.9.2 UNIX and Capacity Planning

UNIX capacity planning is usually done in conjunction with performance measurements. The usual way of dealing with increasing disk requirements is by adding resources rather than optimizing use of the current ones. This is because disks have been growing in capacity and speed faster than processor speed, thus resulting in cheaper disk resources.

Processor upgrades in the past tended to be much cheaper in the UNIX world than in S/390 (this is no longer the case). When an upgrade is not sufficient to carry the increased load, SMP and cluster parallelism are used to increase the processing power available to a given application.

9.9.2.1 Standard Procedures

When a disk fills up, the system administrator just adds another one to the system. This is easy to do since disk arrays are now the standard way of providing a large disk storage capacity. Modern UNIX systems such as IBM AIX can spread a file system on several physical disk drives and be dynamically extended.

Whenever a disk-intensive application saturates the disk drive bandwidth, the usual way of dealing with this situation is to buy a new disk, preferably faster (which may require an upgrade of the disk drive interface), and use filesystems migration tools to put the most frequently accessed files onto a dedicated disk.

The processor requirement is less easy to satisfy. When an application or a database management system saturates its available processing power, you have to find some way of upgrading it. The obvious way is to upgrade the CPU of the machine, but in some cases, it is not enough. You should then consider a parallelization of the application or database in order to run it on a cluster or SMP.

In this case, the parallelization study should also consider high-availability and fault-tolerance issues. When an application is distributed upon separate machine for increasing the processing power, a high-availability facility, such as IBM's HACMP, can provide fault-tolerance at an incremental cost.

9.9.2.2 Tools and services

Many vendors offer capacity planning tools services. IBM, for example, offers an extensive set of capacity planning tools and services (for details, contact your local IBM representative).

9.10 Storage Management

This task is also important because it guarantees to the user and operation people available space for storing and retrieving data, and so on.

9.10.1 OS390 and Disk Space Management

Generally this task is assumed by a storage management administrator who:

- Defines rules and policies
- Implements rules
- Tracks the available space

DFSMS/MVS with its different components provides a superior storage management environment for the S/390 customer. (There are non-IBM tools which can perform portions of the storage management discipline.)

There are IBM components for that purpose:

- DFSMS/dss to duplicate or copy files
- DFSMS/hsm to migrate and retrieve files

Recent techniques that involve data compression have been implemented on the IBM mainframes. These techniques bring some space saving.

Data compression using software has been available for many years, but its use has been impacted by the large overhead incurred in compressing and/or decompressing records. Overheads for light database usage, for example online transaction processing, can be relatively low (5-10%); however the overhead can be prohibitive for heavy database usage (100% or more for data warehouse-type applications).

IBM S/390 Hardware Data Compression substantially reduces the cost of data compression (by a factor of 5-10 times), and can not only lower disk storage costs, but also improve system throughput by enabling more data to be stored in memory, and by reducing the amount of data transferred over channels.

9.10.2 UNIX and Disk Space Management

As noted previously, the disk space management is often done by the system administrator under UNIX.

Most modern versions of UNIX, such as AIX, offer a degree of dynamic disk space management. For example, under AIX, you can increase the size of a file system by adding disk space from other disk drives as they are added to the system. You

can migrate file systems from one physical disk to another. You can mirror or replicate file systems, and you can also copy files to tape and restore them.

Under UNIX, there is no standard way of implementing file migration from disk to tape. Packages are available, but each one requires its own set of operating procedures.

One such package is ADSTAR Distributed Storage Manager (ADSM) IBM's storage management multi-platform product. With ADSM, you can remotely backup and restore files, do unattended regular backups, and maintain a backup storage hierarchy. ADSM also implements Hierarchical Storage Management (HSM), which moves less-used files from disk to secondary storage (for instance, tape or optical devices). ADSM HSM automatically selects files to move according to parameters set by the system administrator (file size and time since last access). What is left on the disk after the migration is a stub indicating the file status and location on storage. UNIX commands such as `ls` can work transparently with this stub. Read or write attempts trigger a file recall from secondary storage. This recall is automatic from the application's point of view.

For more information on ADSM, see the following URL:

`http://www.storage.ibm.com/storage/software/adsm/adfaq.htm`

9.11 Tape Management

Storage management would fall short if it was not accompanied by tape management. In spite of the phenomenal advances in optical storage, magnetic tape is still by far the most prevalent and cheapest data storage media.

9.11.1 OS/390 and Tape Management

OS/390 and IBM System Managed Storage offer a complete solution to the automation of the use and management of magnetic tapes. 3.4.1.2, “Tape Management” on page 57 contains an extensive description of the facilities offered and their benefits for a consolidated server environment.

9.11.2 UNIX and Tape Management

Under UNIX, there is no standard way of managing a tape library. Some packages, such as ADSM (see 9.10.2, “UNIX and Disk Space Management” on page 142), manage tape storage as part of their hierarchical storage management, and can work with commercial third-party devices, like tape silos and automated tape libraries.

9.12 Configuration Management

Configuration management is an important point to consider because the IT system must enable the configuration to change in the following cases:

- When the business increases
- For maintenance purposes
- During recovery situations

Different system components must be taken into consideration, such as processor and associated elements (central and expanded storage, channels and I/O configuration.

9.12.1 Large System Configuration Management

IBM S/390 with OS/390 has, for many years, offered different options so that customers can modify the processor and I/O configuration when required.

9.12.1.1 Logical Partitions

S/390 Servers have a function, known as Processor Resource/System Manager (PR/SM), which enables multiple operating environments to operate on one physical server.

Each logical partition consists of one or more server engines (which may be dedicated, or shared with other partitions), memory, and I/O channels and its own copy of the operating system.

Processor dispatching is based on parameters specified by the installation, and its algorithms ensure that all resources are used efficiently. Logical partition definitions can be varied, and resources re-partitioned between them

Thus logical partition re-configuration management also forms part of the configuration process in an S/390 environment, and procedures and tools exist for this purpose.

9.12.1.2 Processor

This component can be managed in order to change the power a customer needs:

- Partition weights can be dynamically modified in order to add or remove power to OS/390 partitions.
- The VARY command also can be used to add or remove logical processors defined to a partition. This allows CP parallelism for applications that can take advantage of it.
- Central and expanded storage reconfiguration options. Today many OS/390 customers have different partitions dedicated to specific environments, such as production, test, and development. Usually during the night shift only the production environment runs batch-type applications; all others are idle. In order to use more storage to implement data in memory techniques, central and expanded storage can be dynamically allocated to the production environment. This is done with hardware and software definitions and activated by a VARY command from the operator console.

The RMF component of OS/390 allows the use of online monitors for configuration control or event tracking.

9.12.1.3 I/O Configuration Management

This part of the IT system is changing very quickly, since disk capacity and speeds have been growing faster than processor speed.

Since 1990, the I/O configuration has seen an improvement in ease of use. The ESCON architecture now uses optical fiber cables instead of large copper cables. The ESCON architecture also increases the maximum cable length while maintaining good performance. Fiber optic channels allow devices to be connected at distances of up to 43Km.

To change the configuration, the customer needs to have tools to add or remove I/O devices, and control the I/O configuration. To do those tasks, OS/390 offers integrated products such as:

Component ***Function***

HCD

This component is used to modify the I/O configuration kept in IODF files.

ESCON Manager

This component is used to control the I/O configuration. It must be operational in a Parallel Sysplex configuration, because it provides a single image view for each device accessed from all sysplex images.

Hardware Configuration Manager (HCM)

This component offers a graphical interface on a PC to work with HCD. HCM is part of the OS/390 base in V1 R4.

Note: Today the IODF files built with HCD also contain the ESCON Director's (ESCD) configuration data. The configuration data can be read, written and activated from HCD. Because of this, it needs ESCON Manager running in order to access the ESCON Directors.

9.12.2 UNIX Configuration Management

UNIX has no standard configuration management. However, some parallel systems can dynamically add or remove processors from the pool of nodes available for a given workload. Each vendor has its own set of parallel system management tools, although clustering standards are currently being debated.

I/O configuration management is also more limited than under OS/390. With UNIX, you can dynamically increase the size of file systems. You can also hot-plug spare disks in third-party disk farm racks that work with most UNIX machines.

9.13 Problem Management

Problem management directly impacts system availability and reliability.

9.13.1 OS/390 and Problem Management

In an OS/390 environment this task involves the customer administrators.

There are different kinds of problems that need to be managed:

- Hardware problems (processor, I/O and network)
- Software problems (error correction)

9.13.1.1 Hardware Problem Management

Hardware problem management can involve reconfiguration techniques. The problem is to react fast enough in order to avoid an impact to the production system.

In S/390, many processor, memory and channel error situations are handled automatically, and recovery actions are taken by the hardware and microcode, transparently to the running applications, as follows:

- Processor (reconfiguration of CP, storage, and channels)

In the event of hardware failure of a processor unit, the S/390 G3 and G4 servers have the ability to substitute a failing processor unit with a spare one.

In the event of storage problems on a S/390 G3 and G4 server, the administrator can do a power-on reset (POR) on the machine and restart the applications.

- I/O (reconfiguration of paths and addresses)

  The administrator can use commands to prohibit the use of a specific failing unit.

  When a configuration has ESCON Directors, there are different ways of managing the I/O configuration. Channel links can be blocked using the ESCON Director console or the ESCON Manager software product. An alternate path can be used after the failing one has been stopped.

9.13.1.2 Software Problem Management

Software problem management is slightly different from hardware problem management: in most cases, its occurrence can be predictable since it follows some software change or modification. The general software problem management procedure follows this pattern:

- Detection

  You should gather all the elements (such as messages, errors codes and wait codes) needed to inform the IBM or internal support people about the problem.

- Bypass (if possible)

  This is possible when the bypass is simple and fast to apply.

- Correction

  This should be applied on a test environment before the production one.

- Test

  This step is intended to check that the previously detected problem has been corrected, and that the correction has not introduced any new defect.

- Apply

  Put the new version into production.

Today, most customers have an OS/390 test partition to test the software problem corrections.

9.13.2 UNIX and Problem Management

As with OS/390 servers, problem management for UNIX servers also has to consider both hardware and software problems.

9.13.2.1 Hardware Problem Management

In order to deal with inevitable hardware problems, high-availability UNIX systems have been developed, such as IBM HACMP (see 9.5.1, “Hardware Recovery on UNIX” on page 133.) High availability involves both disk and processor redundencies.

Disk arrays (RAID) provide fault-tolerant disk storage. Twin-tailed (or “Y-shaped”) cables can be used to connect a RAID array to two different UNIX systems. Thus,

if one of the processor fails, the other processor can still use the disks under control of take-over software.

9.13.2.2 Software Problem Management

Until recently, there was no such thing as a test partition under UNIX (the Sun UE10000 now offers up to 8 logical partitions). However, relatively inexpensive UNIX machines can be added to a site so that they are always available to test a new software release.

The same cycle as in OS/390 is used for UNIX sites: detection, bypass if possible, correction, test, apply in production. However, test partitions are usually replaced with dedicated systems.

9.14 Change Management

As with problem management, change management is very important because it can impact system availability.

Changes must be planned involving the business part of the enterprise in order to choose the best period to apply changes. It requires some planning agreement by all involved parts (IT and business) regarding both hardware changes and software changes.

9.14.1 OS/390 and Change Management

The following tasks have to be done:

- Identifying the changes
- Evaluating the impact and find the best period to implement the changes
- Obtaining agreement from all involved people
- Warning people of the impending changes
- During the change operation, having all involved persons on site

Many change control problems can be solved with System-Managed Storage (SMS). For example, different HFS file trees can be stored and managed with SMS. However, this method can generate its own specific problems.

Many OS/390 installations have system residence volumes that are not managed with SMS. These non-SMS volumes include the SYS1.LPALIB and the SYS1.PARMLIB libraries.

The BPXPRMxx member of SYSI.PARMLIB library refers to a set of HFS files which is the file tree structure of OS/390 UNIX System Services. If this setup needs to be changed, you can create a new HFS file tree (under SMS) and a new set of system residencies, as illustrated in Figure 39 on page 148.

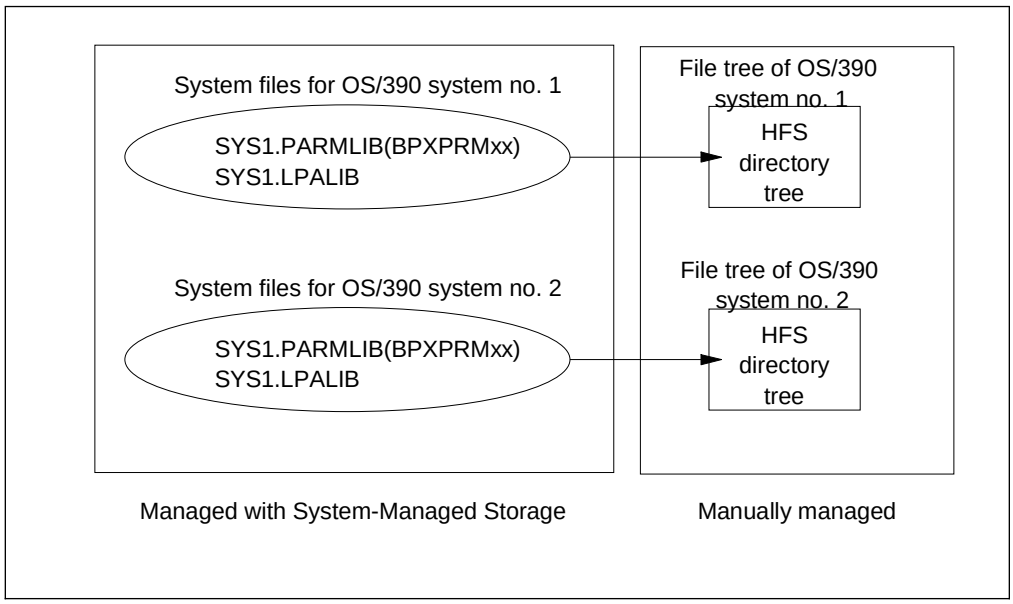


Figure 39. SMS-Managed System Libraries and Non-SMS-Managed HFSs

9.14.2 UNIX and Change Management

Some UNIX systems still have to be managed with a log book as the only tool. Modern systems, however, implement a full change log database, a kind of integrated log book (which should be printed out periodically so that it can be consulted even when the system is down).

For example, AIX offers a configuration database, based on its Object Database Manager (ODM) subsystem. Every time a software update is applied, the files that are updated are first saved to a special directory. Thus, an update can be rejected and the previous versions of the files can easily be retrieved.

Also, the AIX ODM stores the current status of every fileset in the system (the fileset being the finest granularity of installable software in AIX version 4). Commands are available to list the status and version of a given fileset, as well as the version of the whole OS and all its subsystems.

The ODM also stores the current hardware configuration, which is compared to the actual detected hardware subsystems at boot time. When the hardware is modified, any discrepancy between the stored and actual configuration is resolved automatically if possible, and the operator is prompted to accept the ODM updates proposed by the system.

9.15 General Considerations with OS/390 UNIX System Services

Moving to an OS/390 UNIX System Services environment does not bring major changes in the organization. There are still people dedicated to production, system management and applications.

The only element that needs to be added is “people education.” OS/390 people must know UNIX, and conversely UNIX people must have OS/390 knowledge.

9.15.1 IT Organization

The main cultural difficulty is to integrate UNIX-minded people in the different existing teams. The other possible solution is to educate OS/390 people in UNIX. In practice, the size of the installation can determine the approach taken.

The physical organization can also change. A possible organization is to dedicate a separate environment to host the OS/390 UNIX System Services applications separately from the classical OS/390 applications. This could be done easily in a partition which accesses its own HFS datasets.

9.15.2 Skills

With regard to skills, there are different areas to consider:

- System management and operation
- Environmental (for example, the editor)

As mentioned before, the skills of the IT staff have to be improved with appropriate education. For example, on OS/390, the basic element is an *address space* represented by a user, a job or subsystem component. On top of the address space structure, OS/390 UNIX System Services introduces a new element called a *process*, with which UNIX people are familiar.

With OS/390 there are different ways to stop an address space. For example, the CANCEL command stops a job represented by an address space. The UNIX operator uses the `kill` command to stop processes contained in an address space.

Another element that may be different is the security implementation. With OS/390, there are products like RACF that manage the security protecting data sets and files. With OS/390 OpenEdition, a new level of security has been added:

- Access to the user to read an HFS data set
- Access to the user to write an HFS data set
- Authorization to the user to execute a program in HFS data set

Concerning the environmental aspect, there are some differences with the editor. This impacts mainly the developers that use such tools. On OS/390, the traditional editor is a component of ISPF. In order to use it, the user must log on on TSO/E and cannot use this editor through a rlogin session. In fact the user calls the `OS/390 UNIX Servicesdit` OpenEdition OS/390 shell command.

9.15.3 Working Relationships

Considering that the environment is changing for both OS/390 and UNIX people, *teamwork* is of utmost importance. This element is essential because every person involved can get experience on the other environment.

There has been a traditional relationship between operations people and development people in an OS/390 environment. Following consolidation, application developers working on OS/390 UNIX System Services have a different environment for testing applications than they have been used to, and a new understanding with operations will have to develop.

Chapter 10. Language Migration Considerations

The overall experience of UNIX programmers porting to OS/390 is that it is as easy as porting between any other UNIX environments. They are happy to consider OS/390 as a UNIX system. Much of the time spent porting to OS/390 is spent on tasks that are common to porting generally.

This section looks at these generic issues and then goes on to examine issues that are specific to OS/390.

10.1 Generic Porting Issues

In this section we discuss some issues generic to any porting exercise from UNIX to OS/390 UNIX System Services. (Please consult: `http://www.s390.ibm.com/products/oe/` and click on `Porting`, to obtain the most recent guidance on porting applications from UNIX to OS/390 Unix System Services.

10.1.1 Header Files

A *header file* is the name of a library of functions supplied with the compiler. When a programmer invokes certain functions from his C code, he includes the required header files in the code. Thus, portability of code relies upon consistency of header files across different C products.

Unfortunately, this is not always the case and any port between any two UNIX platforms will inevitably involve a certain amount of time locating functions with slightly different names, or that are declared in different header files. The widespread adoption of UNIX95 standards by application programmers will facilitate this process in the future.

10.1.2 Make

The *make* facility is a rule-driven utility for managing the compilation process. The programmer specifies the relationships between programs and make ensures that compiles and links are done correctly and at the right time.

Make implementations differ across UNIX platforms. This means that makefiles themselves need to be ported. That is, code that does not work on OS/390 needs to be identified and changed. The OS/390 make tends to enforce stricter conformance to standards than many other flavors of UNIX, although the differences are smaller with each new release of OS/390.

10.1.3 C Compiler

Similarly, the OS/390 C/C++ compiler tends to require stricter conformance to language standards than some other compilers. For example, differences occur in handling of pointers, prototyping, castings and datatypes.

During the early stages of a port it can be useful to use compiler options such as langlvl=extended, .POSIX: special target, and export_c89_ccmode=1. All of these will reduce the number of compiler error messages.

As the port progresses, these options should be removed if it is desirable to produce the most portable standards-compliant code. A similar technique, useful in advance of a port, is to change the make settings on the source platform to the strictest available.

Some header file, make and compiler differences identified during porting projects are collated in the redbook *Porting Applications to the OpenEdition OS/390 Platform*, GG24-4473 and on the OpenEdition home page (`http://www.s390.ibm.com/products/oe/index.html`), from which the following sections have been extracted.

10.2 Porting Issues Specific to OS/390

This section discusses some porting issues that are specific to OS/390 UNIX System Services (rather than to compilers, and so on).

10.2.1 spawn Versus fork Considerations

If your application creates a lot of processes and you want better performance, `spawn()` is the way to go. Similar to `fork()` and `exec()`, `spawn()` runs much faster and saves resources because it does not have to copy the address space. In fact `spawn()` can optionally place the new process in the same OS/390 address space, even further saving system resources. If your application is multithreaded you must use `spawn()` instead of `fork()`.

If your application is designed to create multiple processes with each running the same program, `spawn()` might not be useful. Many applications rely on having program initialization performed once by the parent process and propagated via `fork()` to all the children processes. The `spawn()` function only propagates a few things like open file descriptors; `spawn()`'s assumption is that the new process will run a different program, not another copy of the same one.

If your application uses pipes or shared memory and you switch to using `spawn()`, you should watch out for the following differences.

10.2.1.1 Applications that Use Pipes

After changing from using `fork()` to using `spawn()`, an application that uses pipes can appear to hang. Often one process will work correctly for a while, then get stuck in a blocking read of the pipe.

A pipe consists of two file descriptors (or fd) such that data written to fd B of the pipe can be read from fd A of the pipe. When a process forks, the pipe gets copied as well. In the most common case, the parent and the child process want to maintain unidirectional communication, for example, the parent writes data into the pipe, and the child reads it. The parent writes data to fd B and the child reads from fd A.

When using `fork()`, the parent and child should normally both close their copy of the unused pipe file descriptor. In our case, the parent should close fd A and the child should close fd B after the `fork()`. So each process uses and leaves open only half of the pipe.

With `spawn()` there is no explicit way for the child to close its unused half of the pipe. Because both ends of the pipe are open in the child process, the child will

never see EOF (end-of-file) on a read of fd A, because the write half or fd B is open in the child. EOF is detected only on a read of fd A when the pipe is empty and all copies of fd B are closed.

The solution is for the parent to mark the file descriptor for its half of the pipe (here, fd B) to be closed-on-exec. If this is done, then fd B will not be open in the child. Since the parent closes its fd B, EOF will be detected in the child after all available data has been read from fd A.

10.2.1.2 Applications That Use Shared Memory

After the change from using `fork()` to `spawn()`, an application that uses shared memory finds that `shmat()` returns -1 if both processes are in the same address space, although this worked when the application used `fork()`. The problem is that when the parent and child processes supply the same shmaddr address on `shmat()` and both processes run in the same address space, the kernel detects that the address range is already in use on the second `shmat()` invocation and denies the request.

If you require the shared memory to be at the same address, you have two choices:

- Run the two processes in separate address spaces and do the `shmat()`, specifying the same starting address.
- Or, when running in the same address space, pass the address of the shared memory from process 1 to process 2. Then in process 2, just use the shared memory and do not do the `shmat()`. If only two processes are involved, then regular memory will suffice, and a `malloc()` in the first process can replace the `shmget()`. Then just pass the address of the heap storage to the second process.

10.2.2 Portable Header Files

If an application on a UNIX system is not POSIX- or XPG4-compliant, then you may not be able to just move it to an OS/390 UNIX System Services system and expect it to compile. Such applications may include headers that are not supported by OS/390 UNIX System Services application services. Porting an application that does not conform to those standards requires that you inspect all headers that may not exist on an OS/390 UNIX System Services system and determine whether or not the application really requires them. As you know, headers can contain all kinds of things, from macros that simply exist for convenience to prototypes of functions that may or may not exist on a particular UNIX system.

Here is a list of some headers that you will *not* find on an OS/390 UNIX System Services system (this list is not complete):

UNIX file	OS/390 UNIX System Services equivalent
<access.h>	OS/390 UNIX System Services equivalent interfaces are in unistd.h, per POSIX and XPG4.
<ar.h>	No equivalent at this time.
<arpa/ftp.h>	No equivalent; you can “borrow” a file from a UNIX system and use it.
<cur01.h>	This header is not standardized; replace it with curses.h.

<dir.h>	OS/390 UNIX System Services supports dirent.h per POSIX and XPG4.
<macros.h>	No equivalent at this time.
<termio.h>	OS/390 UNIX System Services supports termios.h per POSIX and XPG4.
<sys/ldr.h>	No equivalent at this time.
<sys/mntctl.h>	No equivalent at this time.
<sys/mode.h>	This header is non-portable. OS/390 uses modes.h, but the standards do not specifically refer to this header. An include of fcntl.h is more portable.
<sys/param.h>	This header is often unnecessary. Try removing the includes for it and see what falls out.
<sys/ptrace.h>	Although OS/390 does not have this header file, it has a kernel interface (BPX1PTR, see *OS/390 OpenEdition Assembler Callable Services*, SC28-1899). The raison d'être of this callable service is to support the dbx debugger, which was ported from AIX. Much of what AIX's sys/ptrace.h defines shows up in the assembler macro BPXYPTRC. It should be possible to create a header file based on the macro.
<sys/reg.h>	No equivalent at this time.
<sys/vmount.h>	No equivalent at this time.
<sys/vnode.h>	No equivalent at this time.
<usersec.h>	No equivalent at this time.
<userpw.h>	No equivalent at this time.

10.2.3 sys_errlist

Note that if you compile and link a program using
`extern char sys_errlist`,
you will get an informational message:

```
WARNING EDC4011: Unresolved writable static references are
detected.  FSUM3065 The PRELINK step ended with return code 4.
```

However, the link does not halt and you will end up with a program that is executable (c89 or cc returns 0) but will abend when it tries to access sys_errlist. You have to generate a verbose list to find out that it is sys_errlist that cannot be resolved.

This behavior can cause problems when porting, because some packages compile and link a test program to see if they can use sys_errlist. Since c89 or cc returns 0 on the compile and link, it mistakenly appears that OS/390 UNIX System Services supports sys_errlist. To make it known that OS/390 UNIX System Services does not support the external reference, you may have to change a makefile or header file after running a configuration script but before building the package. For instance, the package might define a C macro _HAS_SYS_ERRLIST and you will need to remove that definition. Or the package might have deliberately not defined _USE_STRERROR since it thinks using sys_errlist might be faster; this means you will need to define the _USE_STRERROR macro. These are just some examples;

a package can use other names and methods to determine whether or not it should use sys_errlist.

10.2.4 Dynamic Link Library

Prior to OS/390 Release 3, you need to use the -Wl,dll link-edit option when building a non-DLL application, and just disregard the .x file that is produced. The reason: since you are linking in .o files compiled for creating a DLL, the linkage editor thinks that your application is itself a DLL.

As of OS/390 R3 (make sure you have APAR OW23744), you no longer need to specify -Wl,dll. If you do not specify it, however, you will see a warning message from the linkage editor (which helps to detect that you may have unintentionally exported symbols). By default, c89 keeps going with warning messages, so your executable will still be built.

In order to use DLLs, programs written in C must be compiled with the -Wc,dll option. This option causes different code to be generated, in order to dynamically link to (import) symbols from DLLs. Programs written in C++ are always DLL-enabled, so no special option is needed, and only one style of code is ever generated. Currently, exporting a symbol simply sets a flag in the symbol dictionary entry to indicate that the symbol is exported. So, in C you can create a DLL (which exports symbols) that is not itself DLL-enabled (it does not dynamically link to any DLLs, and so it is itself not compiled with -Wc,dll).

The point here is that what is needed to import and what is needed to export are two different issues. It is the exporting of symbols that causes the link-edit to create a .x file. This is independent of whether any .o files being linked were created from .c files compiled with -Wc,dll.

So, you need not recompile the .c files from your own DLL without -Wc,dll in order to statically link. If all the .c files are compiled with -Wc,dll, but not link-edited with any .x files, then the application is DLL-enabled, but just is not using any DLLs.

You also do not need to recompile without -Wc,exportall (or #pragma export). Prior to OS/390 Release 3, you have to use the -Wl,dll option, but that is the only change you need to make.

10.2.5 ASCII-to-EBCDIC Conversion

The main complaint among UNIX programmers is that OS/390 UNIX System Services uses EBCDIC. This implies that some conversion problems appear during a typical application port.

10.2.5.1 Typical Problem Areas

When porting a program to OS/390, an experienced OS/390 UNIX System Services tester says experience has taught him to keep an eye out for these areas where the ASCII to EBCDIC conversion may cause problems:

- Hard-coded ASCII characters in C code as well as shell scripts

  Avoid using hard-coded values or depending on the values of characters at all costs. For example, a program might use '\012' (octal) instead of '\n'. A program might use characters as indices into arrays that were populated using the ASCII values for indices (for example, in C, if Tab is an array, Tab['a'] is equivalent to Tab[0x61] in ASCII, but not in EBCDIC), and so on.

- Using the high-order bit of a character for some special purpose

  You can do this in ASCII because only 7 bits are necessary for all the printable characters, but that is not true in EBCDIC.

- Assuming the alphabet (a...z) is contiguous

  This is true in ASCII, but not in EBCDIC where there are three noncontiguous groups of letters. Even seemingly harmless code like the following probably needs to be changed:

  ```
  char c;
  for (c='a'; c<='z'; c++)
  { ... }
  ```

- Using code generated by lexx or yacc

  Often, packages contain C code that was generated by the lexx or yacc utilities. This code will probably contain ASCII dependencies and will not work on OS/390. The code needs to be generated on OS/390 by rerunning the utilities. Note that this may introduce EBCDIC dependencies, making the code less portable to other systems, but at least it will work on OS/390.

  For example, y.tab.c is typically generated by yacc and there should be commands in the package's makefile instructing make how to invoke yacc to rebuild y.tab.c. There should also be a comment in y.tab.c that specifies the source file that yacc processed to generate y.tab.c.

- Applications that talk to arbitrary remote systems via sockets (such as an FTP client)

  These applications typically have to assume that all text they receive is ASCII and they send out all text as ASCII. They have to convert the data locally on the fly.

  Consider an FTP client: a user will type a command such as

  ```
  dir foobar
  ```

  The FTP server does not want to see these characters in EBCDIC, so the client must convert the data to ASCII before they are written to the socket. Likewise, if you simply write the data coming from the server to the user's screen, it will be meaningless because it will be in ASCII. The client must first convert the data to EBCDIC. This is true even if the server is running on an EBCDIC system, such as OS/390 or VM.

  However, you must be careful to convert only text data. Some applications may mix binary and text in a data stream. For instance, the server might send 2 bytes of binary data preceding a block of text to represent the number of bytes in the block, a cksum value, and so on.

- Code that relies on byte order of data may not be portable. PC systems are “little endian” (that is, the leftmost byte is the most significant), however OS/390 and most UNIX systems are “big endian.” This typically affects integer and floating point data. If an application is responsible for transferring such data between platforms, you need to either (1) write data exchange logic or (2) translate to text, transfer as text, and then recreate as binary.

10.2.5.2 Functions That Support ASCII Input/Output

Standard I/O and String Library Functions: The OS/390 C/C++ run-time library functions support EBCDIC characters. The new libascii package and the C/C++ compiler version V1R3.0 supply a predefined macro, __STRING_CODE_SET="ISO8859-1". This macro provides an ASCII-like application environment on OS/390. You can download our libascii package, which provides an ASCII interface layer for some of the most commonly used C/C++ run-time library functions. libascii supports ASCII input and output characters by performing the necessary `iconv()` translations before and after invoking the C/C++ run-time library functions. The __STRING_CODE_SET="ISO8859-1" predefined macro generates ASCII characters, constants, and strings.

Converting text files in an archive: You can set an environment variable in your .profile to handle conversion from ASCII to EBCDIC for text files contained in archives. Here is an example showing how to set an environment variable called A2E and then use it:

```
$ export A2E='-o from=ISO8859-1,to=IBM-1047'
$ pax $A2E -rzf foobar.tar.Z
```

“The -o option is not pretty to look at, but once you hide it in a variable, it is easy to use and works perfectly. I have converted millions of bytes of text data this way and have not had a single conversion problem,” says an experienced OS/390 UNIX Services tester.

10.2.5.3 Commands and Functions That Handle Conversion

There are shell commands, TSO/E commands, and C functions that handle ASCII to EBCDIC conversion.

Here are two shell commands that are useful:

- `iconv`

  For example, the command:
  `iconv -f IBM-1047 -t IS08859-1 words.txt > converted.txt`
  converts the file words.txt from the IBM-1047 code set to the ISO 8859-1 code set and stores it in the file converted.txt.

- pax

  For example, the command:
  `pax -wf testpgm.pax -o to=IBM-1047,from=ISO8859-1 /tmp/posix/testpgm`
  backs up the /tmp/posix/testpgm directory, which is in the character set CP1047, into an archive file that is targeted to an ASCII character set (ISO8859-1).

The TSO/E commands OPUT, OGET, and OCOPY let you convert files between ASCII and EBCDIC.

The C functions `__atoe()`, `__atoe_l()`, `__etoa()`, and `__etoa_l()` also perform ASCII-EBCDIC conversion.

10.2.6 Supported Compilers

c89, cc, and c++ with OS/390 UNIX System Services support these compilers:

- SAA AD/Cycle C/370 Version 1 Release 2, installed with Language Environment Version 1 Release 3 or higher

  For OS/390 OpenEdition Release 1 and Release 2, this is the default compiler used by c89 and cc.

- IBM C/C++ Version 3 Release 1, installed with Language Environment Version 1 Release 4 or higher

  Only the C compiler is supported under OS/390 UNIX System Services. The C++ compiler is not supported. However, if you compile with C++ V3R1 outside of OS/390 UNIX System Services, you can use the object it creates under OS/390 UNIX System Services, so long as you have the level of OS/390 UNIX System Services that supports C++ and DLLs. That level is:

  – MVS/ESA 5.2.2 or higher

  – The following with PTFs (SPEs) for C++ and DLLs: Language Environment Version 1 Release 5, Shell and Utilities (FMID HSU1130 or higher), dbx (FMID HDX1130 or higher)

- IBM C/C++ Version 3 Release 2, installed with Language Environment Version 1 Release 5 or higher

  Both C and C++ are supported. For OS/390 Releases 1 and 2, this is the default compiler used by C++ (cxx).

  OS/390 UNIX System Services support for C++ and DLLs is available beginning with the configuration listed under “IBM C/C++ Version 3 Release 1.”

  The pre-AD/Cycle non-LE compilers are not supported for use under OS/390 UNIX System Services.

  OS/390 Release 2 includes a new release of Language Environment and C/C++ compiler. This is the follow-on to C/C++ for MVS/ESA V3R2, and includes compile-time and execution-time performance enhancements such as:

  – Precompiled header file support (PCH) for improved compilation time for C and C++

  – The OPTIMIZE(2) compile-time option, to perform global optimizations for improved execution time for C

  – The IPA compile-time option, to perform interprocedural analysis optimizations for improved execution time for C

10.2.7 IMS Batch Monitor Program

You can run a C program under an IMS BMP and have it call OS/390 UNIX System Services. We specifically had to change dub so that it supported that environment. In the BMP, IMS LINKs to the C program, which gets dubbed and does its OS/390 UNIX System Services processing. When the program terminates, Language Environment (LE) calls undub so that the next transaction running in the BMP does not have any leftover OS/390 UNIX System Services environment. In order to debug this program with dbx, you will need to “dbx attach” to the program once it starts. This might not be easy to do for a program of short duration. You may need to add a sleep or SIGWAIT to the program to allow yourself time to

perform the dbx attach. Once the attach completes, you can send a signal to the program to get it running again.

10.2.8 libascii: Supporting ASCII Input/Output

10.2.8.1 Overview

The libascii package helps you port ASCII-based C applications to the EBCDIC-based OS/390 UNIX System Services environment.

The C/C++ run-time library functions support EBCDIC characters. The libascii package provides an ASCII interface layer for some of the most commonly used C/C++ run-time library functions. libascii supports ASCII input and output characters by performing the necessary iconv() translations before and after invoking the C/C++ run-time library functions. Note that not all C functions are supported (see 10.2.8.4, “Limitations” on page 161 for additional information).

The OS/390 V1R3.0 C/C++ compiler predefined macro _STRING_CODE_SET__="ISO8859-1" generates ASCII characters rather than the default EBCDIC characters. Using this with the libascii code provides an ASCII-like environment.

The libascii package is as thread-safe as the run-time library except where stated under 10.2.8.4, “Limitations” on page 161 below.

You can download the package. The libascii source code is licensed by IBM; for more information, see the readme file in the package. Note: If your browser does not work with this link, use the following URL instead:

`ftp://ftp.s390.ibm.com/u/ftp/os390/oe/toys/libascii.tar.Z`

To report problems or ask questions, send e-mail to `libascii@vnet.ibm.com`.

10.2.8.2 Floating Point Conversion

The libascii archive also contains four functions to convert between IEEE floating point and S/390 native hex floating point. The OS/390 V1R3.0 C/C++ compiler does not support IEEE floating point. Math operators such as + (add) and / (divide) and run-time library functions such as `printf()` and `sin()` that use IEEE floating point numbers will produce undefined results. The main difference between the two floating point formats is that IEEE supports a larger range of numbers, up to 10 to the 308 power. S/390 supports better precision but a smaller range, up to 10 to the 75 power.

The four floating point conversion routines included in libascii are:

- `void ConvertFloatToIEEE(void *source, void *destination);`
- `void ConvertDoubleToIEEE(void *source, void *destination);`
- `void ConvertIEEEToFloat(void *source, void *destination);`
- `void ConvertIEEEToDouble(void *source, void *destination);`

10.2.8.3 Installing the libascii Code

The libascii tar file contains all the parts required to create the libascii archive. To install:

- Create a directory where you want libascii installed.
- Copy the tar file to that directory.
- Unwind the file using this command:
  `tar -xvfoz libascii.tar.Z`

The libascii source files, makefile, and readme file should now be installed.

Building the libascii.a archive file: You can use the makefile file provided to build libascii.a from the libascii source files.

To build libascii.a, cd into the directory containing the libascii source files and issue the make command.

Building and running the samples: After you have made the libascii.a archive file, cd into the samples directory and issue the `make` command. This builds the samples.

To run the sample programs, issue these commands from the samples subdirectory:

```
sample1
sample2
```

Using libascii: After you have installed libascii and built the archive file, you need to do the following to use the libascii functions:

1. Make minor modifications to your C source code and recompile, using the __STRING_CODE_SET__="ISO8859-1" predefined macro.
2. Link-edit your application with the libascii.a archive file.

   First, you need to determine which parts in your application will be compiled with `-D__STRING_CODE_SET__="ISO8859-1"` specified to generate ASCII strings. It is possible that part of the application will be compiled to generate ASCII strings while other parts will be compiled to generate EBCDIC strings.

For all parts compiled with -D__STRING_CODE_SET__="ISO8859-1", use libascii. The following steps describe what to do with a program that will use the libascii functions:

1. Find all the C/C++ functions that require EBCDIC input (for example, `fopen()`) or produce EBCDIC output (for example, `readdir()`). Both the `nm` shell command and the following `grep` command will help perform this task.

   This `grep` command displays the file names that contain a libascii function followed by the character "(":
   `grep -l -f /the-libascii-directory/libascii.lst *.c`
   where `the-libascii-directory` is the path of the directory containing the libascii package.
2. Determine which C/C++ functions were obtained are not supported by libascii. (See 10.2.8.4, "Limitations" on page 161 for additional information.) For any unsupported functions, you will need to either add functions to libascii or

change the code to add `iconv()` ASCII/EBCDIC conversions around the run-time library calls.

3. Make sure all the required header files are included, as specified in *OS/390 C/C++ Run-Time Library Reference*, SC28-1663. For example, `strncasecmp()` requires <strings.h> to be included.
4. Add the statement `#include "_Ascii_a.h"` to your source file, after all the existing `#include` statements.
5. Recompile using the -D__STRING_CODE_SET__="ISO8859-1" option. It will make the compile generate all strings defined in your program in ASCII rather than EBCDIC format.
6. Link-edit your application with the libascii.a archive file.

10.2.8.4 Limitations

The libascii interface package is code that we found useful in our IBM porting work, and it is offered “as is” for your use. It is intended to assist in getting your applications running as quickly as possible.

These are some of the known restrictions:

- Not all the C functions are supported. The file libascii.lst contains a list of supported ASCII run-time library routines.
- For some of the supported functions there are known restrictions, as follows:
  - `getopt()`:

    The libascii `getopt()` function is not thread-safe. The second argument is changed for a short period of time from EBCDIC to ASCII and then back to EBCDIC.
  - The `printf()` family of functions:
    - The `%$n` specification is not supported in the format string.
    - They are limited to 2048 bytes of output. To increase the size of the strings and output supported, change `#define MAXSTRING_a` in global_a.h.
  - The `scanf()` family of functions:
    - The `%$n` specification is not supported in the format string.
    - The maximum number of arguments supported is 20.
    - They are limited to 2048 bytes of input. To increase the size of the strings and output supported, change `#define MAXSTRING_a` in global_a.h.
- The interface layer is not NLS-enabled; it only supports conversions between the ISO8859-1 and IBM1040-1 character sets.

10.2.9 Porting with pthreads

This list of “differences” encountered with pthreads and mutexes on OS/390 UNIX System Services was compiled by customers who subscribe to the OE-MVS mailing list:

Most of these differences exist because OS/390 UNIX System Services implemented the POSIX.4a draft 6 standard rather than the final version, POSIX.1c

draft 10. The book *Pthreads Programming* by Nichols, Buttlar, and Farrell (ISBN 1-56592-115-1) has a chapter on these differences.

10.2.9.1 Thread-Safe Variants of POSIX Routines

The pthreads standard defines thread-safe variants of existing POSIX functions (for example, `strtok_r()` instead of `strtok()`); however, these are not available under Open Edition. IBM's response is that under OS/390 UNIX System Services, the normal versions are thread-safe so you can use them directly. Because the two variants have differing prototypes, this is a problem if you are porting code that contains the thread-safe variants. You have to either change the code or provide your own versions of the _r routines, which map onto the “normal” ones.

Here is an example of how to use a macro to have your code call the normal version of the `strtok()`function instead of the thread-safe version:

```
#define strtok_r(s,sep,lasts) strtok(s,sep)
```

10.2.9.2 Static Initialization of mutexes

OS/390 UNIX System Services does not allow you to statically initialize mutexes with `pthread_mutex_initialiser()`. To initialize mutexes, use `pthread_attr_init()` and `pthread_mutexattr_init()`. In addition to those two, you may want to use `pthread_mutexattr_setkind_np()`.

10.2.9.3 pthread_delete_key()

OS/390 UNIX System Services does not provide the `pthread_delete_key()` function.

10.2.9.4 pthread_attr_setdetachstate()

OS/390 UNIX System Services does not define `pthread_create_detached()`, and the call to `pthread_attr_setdetachstate()` is slightly different, so look at the interface. You can set a variable to `__detatched` and use this on the call instead.

10.2.9.5 Defaults for detachstate

For `pthread_attr_setdetachstate()`, OS/390 UNIX System Services and other platforms vary in their defaults for `detachstate`

10.2.9.6 Process-Shared Attribute for mutexes

OS/390 UNIX System Services does not support mutexes shared across processes. You can use semaphores instead.

10.2.9.7 pthread_getspecific()

`pthread_getspecific()` has a slightly different prototype under OS/390 UNIX System Services than that specified in the standard.

OS/390 UNIX System Services provides two forms of pthread_getspecific:

- `pthread_getspecific()`
- `pthread_getspecific_d8_np()`

10.2.9.8 Value Returned on Error

When OS/390 UNIX System Services pthread functions encounter an error they return -1 and set `errno`. Other platforms return the error number as the function value. For example, `pthread_mutex_trylock()` returns -1 and `errno` contains `ebusy` when the lock is occupied, instead of the POSIX-specified behavior to return a value of `ebusy`

10.2.9.9 Using a C++ Function Pointer

If you are using C++, you cannot use a C++ function pointer in calls to `pthread_create()` and `pthread_cleanup_push()`. This is also a problem for `qsort()`, `atexit()`, `bsd_signal()`, and for defining signal catchers, `signal()` and `sigaction()`. Compilers on some platforms do not discourage mixing C++ and C functions, which allows undesirable programming practices such as trying to use a C++ function when invoking `pthread_create()`. You cannot solve this problem with casting. If you want to invoke any RTL function that receives a function pointer, you must specify a C function.

This is not specifically a pthread issue--this limitation applies to all library C functions.

10.2.9.10 sigwait

OS/390 UNIX System Services implements sigwait as:

```
int sigwait(sigset_t *set);
```

The standard specifies:

```
int sigwait(sigset_t *set, int *sig);
```

OS/390 UNIX System Services returns the signal that interrupts the `sigwait()` function as a ReturnValue; the standard has it being returned in `*sig`. So `sigwait()` has different prototypes under OS/390 UNIX Services and in the standard. The confusion over which prototype to use extends to other platforms.

10.3 Coexistence Problems

Coexistence problems arise when you are porting to a single OS/390 UNIX Services a collection of programs that previously ran on several separate UNIX machines. Most of the well-written programs do not mind sharing a machine with other programs or other instances (that is, copies) of themselves. It may happen, however, that some of the ported programs cannot run together on the same UNIX system. These troublesome programs fall into two categories:

- Hog programs that think they are the only application in the system and cannot run with most other programs
- Unsociable programs that do not tolerate the presence of another instance of themselves in the same machine

10.3.1 Hog Programs

Hog programs are pieces of badly written code that think they are alone in the system. They are generally written by people who have a very shallow background in UNIX. They can also be programs that have been ported straight from the PC world (generally from DOS or Windows), with a simple recompilation, and labeled “ported to UNIX” by unscrupulous vendors. This especially happens with C programs. After all, it is C code, and we all know that C is portable, right?

The typical hog symptoms involve:

Symptom Nickname	***Description and Remedies***
CPU hogging	Consuming a high percentage of CPU time for a task that can easily be done with other CPU-friendly methods. A redesign of the code may be needed. Use UNIX performance tools to find the bottleneck.
Tight polling	A special case of CPU hogging. The program polls (that is, continually reads) a resource in tight loops and consumes a high percentage of CPU time for doing nothing. A typical example includes keyboard polling, non-blocking reads on files without waiting, etc. Recode with UNIX blocking system calls and, if not possible, loops that call `sleep()` and `yield()`.
Mono-user-mindness	Locking a resource (file or device) without imagining that other applications might want to use it. The cure might be as simple as adding semaphores and UNIX locks, or it might require a redesign.
RAM hogging	Certain DOS programs ported to UNIX do a `malloc()` on a large memory area. How large? Well, they first call system-dependent routines to determine the amount of real (RAM) storage available, and then allocate most of it. This is totally useless under UNIX, especially under OS/390, since disk storage has various levels of cache, and virtual memory is designed to avoid this in the first place. Recode with a reasonably sized `malloc()`, a memory-mapped file, or read/write to an OS/390 dataset in extreme cases.
Monothread design	In some cases, a program is a CPU hog because it is constantly checking for events to happen from multiple sources (maybe because no reasonable multitasking could be implemented in the program's original environment, or the coder was just not familiar with UNIX.) The result is a program that continuously checks for events by doing non-blocking reads or polling a large number of resources. When porting to OS/390 UNIX System Services, the cure is to recode the event-checking loop with as many different threads as there are event sources. Block the threads on an event source, and use an IPC to inform the main thread that the event has arrived. Also check if a `poll()` system call can help.

All these remedies involve at least a partial rewrite. If this is not possible, you should isolate the hog program in its own OS/390 UNIX System Services partition. It will cost storage and administration overheads, but the CPU waste will be kept reasonable by reducing the CPU portion available to that partition.

10.3.2 Unsociable Programs

Unsociable programs are pieces of code that behave quite nicely with other programs, but assume that only one instance of themselves exists on the system. For example, an unsociable program will use a configuration file with a fixed name, or it will create resources under a fixed name. The assumption is that no other application will use it, and that only one application of this kind runs on the machine.

But when you start consolidating several UNIX servers onto a single S/390, you may find that your OS/390 UNIX System Services machine needs to run several instances of these unsociable programs that were previously leading a hermit life on separate systems, thus generating problems.

The typical unsociable program symptoms involve:

Symptom Nickname	***Description and Remedies***
Fixed naming	The various resources used by the program have a fixed name, and several instances of the program will compete for these resources. Recode to use parameters, either passed as command-line argument or in configuration files (which should support separate instances!).
Port hogging	The program has been designed to listen to a given TCP port, and other instances would return a “port busy” error. Either use the multiple TCP/IP stacks feature of OS/390 UNIX System Services, or change the program into a proper multi-threaded TCP socket server that listens to a port continuously and passes the connections to a new thread on another, dynamically allocated port.
Enforcement of uniqueness	Sometimes, the unsociable program actually enforces the interdiction of launching more than one instance of itself. It can use a lock file, and thus needs to be rewritten. It can also be under control of AIX Resource Control System, (RCS) with a parameter specifying that only one instance is allowed. Use the OS/390 UNIX System Services RCS component to allow several instances.

10.3.2.1 An Example of an Unsociable Program

Case in point: A serial line supervisor program in C running on a PC. It was written in-house by a customer to implement a certain high-level supervision on serial lines. It was well behaved and did not fall into the typical pitfalls of the hog program. But due to its design, it could only handle a limited number of serial lines. The status of each supervised line was shown on a graphic console, and the available screen space (“screen real estate”) of the console was limited. The screen had only space for 32 serial lines, which was more than enough at the time the program was designed. Of course, a few years later, the growth of the customer had generated the need for supervising about 150 lines (remember the famous line dating back to early 8-bit microcomputers: “Nobody will ever need more than a 64KB RAM.”).

Designing, implementing and debugging a new user interface would have been long and expensive (not to mention that the customer's operators would have needed to be retrained). In order to increase the number of supervised serial lines, the customer had to install the program on several machines, each monitoring a number of serial lines and showing their status on a console. The drawback was that the operator had to watch several screens.

Now, when this program was ported to OS/390, the customer also wanted to consolidate the various alarm and status consoles spread in his control room.

Hence, he decided to port the serial line supervision program to OS/390 UNIX System Services, with as little change as possible.

It was noticed that this program could display the status of its lines on any X11 display. The solution was to run several copies of the same program on the OS/390 UNIX System Services machine and have as many graphic displays or windows as there were programs. Unfortunately, the program used a given fixed file name, and several instances of the programs were using the same file, leading to conflicts.

10.3.2.2 Fixing the Unsociable Program

The solution was to slightly rewrite the program so that the file name prefix could be passed as a command line argument.

Specifically, the rewrite coders had to watch for functions using file names. They changed these functions to use a command line argument as the prefix for the file names. This way, different instances of the program could use different file names with a prefix given at run-time, as a command line argument.

But the coders found that some Inter-Process Communications (IPC), such as communication queues, also used fixed names or values, leading to the same kind of conflicts. The program used the `ftok()` function to compute the ID of IPCs. This function takes a character and a file name as parameters and produces a key for opening an IPC. The same character and the same file name in all the instances of the program produced the same key, resulting in a resource clash. The solution was to change the `ftok()` calls to use another name passed as a command line parameter.

10.4 COBOL Applications

The OS/390 COBOL compiler is not supported on OS/390 UNIX System Services.

Currently, the two main COBOL languages on the UNIX platforms are :

- ACUCOBOL
- Microfocus COBOL

They are very similar and have the same extension to handle the screen I/O operations. This extension is not supported by the native OS/390 compiler. The parts of the program using the extension facilities have to be rewritten.

There is a plan to port the ACU Compiler to OS/390 UNIX System Services.

10.4.1 Using a COBOL Load Module from OS/390 UNIX System Services

A COBOL OS/390 load module can be run from OS/390 UNIX System Services (for example, called by a C program).

The COBOL module can call a C program (for example, to read an OS/390 data set), get parameters (for example, using OS/390 cross memory) and start a process (for example, accessing an Oracle database).

10.4.2 Hints

In the OS/390 development group, a COBOL development environment has been established within the OS/390 UNIX System Services shell for UNIX-experienced COBOL developers. It is a mostly automated hybrid solution with a makefile and a few C language utilities for moving things between OS/390 and HFS, submitting jobs and viewing the output directly from the OS/390 UNIX System Services shell. The work was done principally by an Independent Software Vendor (Open Software Associates, Australia).

The developer states:

> You run the OS/390 COBOL compiler to create an object file, which can then be linked into an OS/390 UNIX System Services executable. The compiler can do all its I/O on OS/390 UNIX System Services HFS files: source, object, listing. We have a separate program to incorporate any COPY code into the source files before compiling, as we have not worked out any way to include COPY code directly from HFS directories.
>
> The source file from the HFS must contain 80 characters per line, including any EOL characters, so we have written a fixcob.c program to do that before compiling. Since the OEDIT editor removes trailing spaces from the ends of lines, you must rerun fixcob each time you edit the source.
>
> We have a standard JCL template for generating the JCL file that invokes the OS/390 compiler. We first copy the JCL file to an OS/390 partitioned dataset and then submit the job from OS/390 UNIX System Services.
>
> If the job completes successfully, the object file will have been created and we use the OS/390 UNIX System Services linker (binder) to create an OS/390 UNIX System Services executable.
>
> Most of these steps are automated (see Makefile). Unfortunately, you must first run make with a .o file target and then, when you are notified that the job has completed, you can make the executable. We have not been able to get make to wait for job completion. It continues as soon as the job has been submitted.
>
> When you run the job under OS/390 UNIX System Services, you can call a C utility to dynamically allocate the file to a DD name referred to in the COBOL file SELECT statement. We have sample programs to demonstrate this and also one to demonstrate receiving command line arguments from the OS/390 UNIX System Services shell.
>
> Following is the Makefile:

```
LD = c89
SUFFIXES: .c .cbl .cbf .o .exe
#  Default target
all:  cname.exe
# Pad all lines out to 80 chars
.cbl.cbf:
    fixcob < $? > $@
#  Submit compiler job; "make" can't tell when it's done
#  The JCL tells the compiler to read from $*.cbf, creating the object
#  file $*.o and the listing $*.cbf.lst
.cbf.o:
```

```
  sed -e 's/XXINPUTXX/$?/' -e 's/XXOUTPUTXX/$@/' < $*.jcl > $*.jcl.make
  tso -t "oget '$*.jcl.make' 'P390D.JCL.CNTL(MAKE)'"
  tso -o "submit 'P390D.JCL.CNTL(MAKE)'"
#  Use this rule to build ddalloc.o .c.o:
  cc -D_ALL_SOURCE -c $?
#  Use OS/390 UNIX System Services linker to make OS/390 UNIX System
Services executable from object files .o.exe:
  $(LD) $(LDFLAGS) -e $* -o $@ $? ddalloc.o $(CLIBS)
```

The fixcob.c program could be posted as follows:

```
#include <stdio.h>
main()
{  char buffff81“; while(fgets(buf, 81, stdin) != NULL)
  { int i = strlen(buf); while(i-- > 0)
    {  if (!isspace(buffffi“))
       { i++; break;} }
   memset(&buffffi“, ' ', 80-i); fwrite(buf, 1, 80, stdout);
} }
```

The JCL template looks like the following:

```
//OECOBOL  JOB (),CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1),NOTIFY=$SYSUID
//COMPILE  EXEC PROC=IGYWC,
//     PARM=(RENT,TRUNC(BIN),OFFSET,PGMN(LM),TEST(NONE,SYM))
//*  Listing goes here
//SYSPRINT  DD PATH='/u/p390d/XXINPUTXX.lst',
//          PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//          PATHMODE=(SIRWXU,SIRWXG,SIROTH)
//*  COBOL source file
//SYSIN     DD PATH='/u/p390d/XXINPUTXX',PATHOPTS=(ORDONLY)
//*  Object file
//SYSLIN    DD PATH='/u/p390d/XXOUTPUTXX',
//       (copy the PATHOPTS and PATHMODE from SYSPRINT above)
```

The preceding sample programs are to be used “as is,” and are not warranted by IBM or Open Software Associates.

Chapter 11. Communications Considerations

This chapter underlines the differences between regular UNIX and OS/390 UNIX System Services in network communications. The OS/390 UNIX System Services implementation of TCP/IP is highly unusual from the point of view of a UNIX programmer, yet the normal services (Telnet, FTP, and so on) are available, and application portability is assured by the availability of all the standard TCP/IP APIs (sockets, XTI.) A few services have special features.

11.1 TCP/IP Differences

When you come from the “small box” UNIX world, the OS/390 UNIX System Services implementation of TCP/IP is likely to surprise you. Following are the most striking differences.

- OS/390 UNIX System Services TCP/IP uses the notion of virtual IP address, which is not tied to a specific network adapter.
- You can have as many as eight different TCP/IP stacks, instead of just one, each with its own characteristics, address resolution, and possibly IP address.
- Some services are not managed exactly as under regular UNIX.

11.2 Virtual IP Addressing (VIPA)

Virtual IP Addressing (VIPA) is a way to isolate a TCP/IP application running on S/390 from any failure of a physical network adapter. To use VIPA efficiently, your S/390 should have several network adapters. Each network adapter has its own address, but the TCP/IP application only uses the virtual address.

Note: You can even have several virtual addresses, each corresponding to several physical adapters. But in order to simplify, we will just talk about one virtual address here.

If an adapter fails using a non-VIPA (classical) addressing, all the TCP connections opened using this adapter's address will time-out and fail, possibly requiring some applications to be restarted. With VIPA, the TCP connection does not know the address of the adapter. It is merely routed through whatever adapter is available. If a given adapter fails, the normal TCP/IP routing protocols of the surrounding network will mark this interface as bad and will route TCP/IP packets through one of the remaining adapters.

But won't the applications still have to be restarted? No. The TCP protocol is connection-oriented, and tries to send any packet that has not been acknowledged. This is true both for outbound (OS/390-to-external world) and inbound (external world-to-OS/390) packets. Thus, when an adapter fails, the TCP layer tries to send the packets lost during the transition, and a new route is established within 15 to 30 seconds. The TCP applications normally do not notice anything except a momentary slowdown, since all retransmissions are handled in the TCP/IP layer, not by the applications.

If you are familiar with IP routing, you probably noticed that this is similar to having a router embedded into the S/390 and making a certain IP address (the virtual address) reachable through several routes (the network adapter addresses.)

Of course, the network to which the S/390 is attached should be configured accordingly. Specifically, no application should use the adapters' addresses, and no name server or hostname table should contain it. All the applications should use the virtual address.

The TCP/IP component of OS/390 should also be configured properly. The VIPA scheme assumes that the routes are automatically updated if an adapter fails. Hence, in order to use VIPA, you should set up and configure a dynamic routing program such as `routed` on the OS/390 system.

11.2.1 Benefits

- Fault tolerance, since an adapter failure is not noticed by the applications.
- Easier operations, since you can have redundant links and can take down one of those without a warning.

11.2.2 TCP/IP Routing and VIPA

When you configure TCP/IP on an OS/390 UNIX System Services system, you can choose between static routing and dynamic routing. Static routing only makes sense if the system is in a very stable network, with few or no changes, and if it does not have to act as a TCP/IP router. That might be the case if the S/390 machine is connected to a LAN where all the outside connections are routed by another machine.

Dynamic routing means that you run a program on the OS/390 UNIX System Services machine to act as a router. More precisely, this program is able to understand at least one of the routing protocols used on TCP/IP networks. A router usually has several network adapters. The routing program maintains routing tables that are updated according to the routing messages it receives through the various adapters.

The most basic routing program is Routed. It understands only the oldest routing protocol Routing Internet Protocol version 1 (RIPv1). This is the only routing program available in OS/390 UNIX System Services. Some UNIX systems can run other programs such as Gated, which understand other, more recent, protocols.

But since you would not want to turn your OS/390 UNIX System Services machine into a real router (a dedicated machine would be much more cost-effective), this is not a real limitation.

If you use VIPA, you probably want to have automatic recovery of network adapter failures (with, of course, several network adapters). As we mentioned in 11.2, “Virtual IP Addressing (VIPA)” on page 169, this implies that you configure Routed on the OS/390 UNIX System Services system. This allows the IP packets to/from the virtual IP address of the OS/390 UNIX System Services system to be routed to the physical address of another adapter.

11.3 Multiple TCP/IP Stacks

Starting from TCP/IP V3R1, it is possible to have up to eight independent TCP/IP stacks on an OS/390 UNIX System Services system. The TCP/IP applications see the different stacks as separate hosts, each with its own IP address. Each TCP/IP stack in the machine runs its own code in its own address space, so you can even mix different versions of TCP/IP for OS/390 on the same processor, as long as they are V3R1 or later.

Stacks are identified by the TCPIPJOBNAME parameter in the TCPIP.DATA data set. An OS/390 UNIX System Services application can get the value of the current stack or select a new stack with the `getibmopt()` and `setibmopt()` calls.

11.3.1 Multiple Stacks Configurations

The most obvious way of configuring multiple TCP/IP stacks is to connect each stack to its own network adapter.

Each TCP/IP stack maintains its own routing layer and other state information. This can increase the storage consumption. In order to reduce the storage requirements, you may choose to have only one stack connected directly to a network adapter. The network-connected stack is known as the *front-end* stack. The front-end stacks then communicate with other stacks, known as *back-end* stacks, though IUCV connections. The Inter-User Communication Vehicle (IUCV) is an OS/390 facility allowing communication between different address spaces.) This keeps all the routing and network state information in the front-end stack only. Note that the front-end and back-end stacks can be on different processors if you have a Parallel Sysplex machine.

11.3.2 Multiple Stacks and VIPA

Running multiple stacks and multiple adapters while maintaining the illusion of a single virtual IP address for the applications gives you added fault-tolerance and operational benefits:

- Easier operation: You can test a new version of applications with a stack and have production applications running on another stack. This allows performance measurements of a new application to be very precise and to avoid interference from production software.
- Fault tolerance: You can have a back-end stack with a virtual address connected to several front-end stacks. These stacks can be on other processors if you have a Parallel Sysplex. A single failing network interface or even TCP/IP stack abend cannot bring the links down. Applications are rerouted in typically 15 to 30 seconds and keep running.

11.4 Differences in Standard TCP/IP Services

This section describes differences in the way standard TCP/IP services handle file transfers and sendmail.

11.4.1 File Transfer Protocol (FTP)

The OS/390 and regular UNIX implementations of File Transfer Protocol (FTP) show some differences. The main differences are that:

- You can use FTP to submit MVS jobs to an OS/390 system.
- You can add extra security controls to an OS/390 FTP server.

11.4.1.1 FTP for MVS Jobs Submission

You can use FTP to automate the submission of MVS jobs to an OS/390 system. This is done through the FTP Job Entry Subsystem (JES). Using FTP commands, you can submit jobs, list the status of submitted jobs, receive output of completed jobs, or delete jobs.

From any FTP client, you can enter the FTP JES mode with the subcommand `SITE filetype=jes` (SITE is the FTP subcommand for site- or machine-specific commands).

Once you are in the JES mode, you can:

- Submit jobs with the `PUT` subcommand. The specified file should be a JCL file. It will be sent directly to the JES reader.
- List the status of submitted jobs with the `DIR` subcommand. This will show you pseudo-files containing jobs submitted (input) or results (output).
- Receive output of completed jobs with the `GET` subcommand. Use the `DIR` subcommand to list output pseudo-files corresponding to job results.
- Delete submitted jobs with the `DEL` subcommand. Use the `DIR` subcommand to list input pseudo-files corresponding to submitted jobs.

11.4.1.2 FTP Extra Security Checks

The security of the OS/390 FTP server can be reinforced by user exits (a `user exit` is a point in an IBM-supplied program at which a user-supplied routine may be given control).

The FTP server supplied with TCP/IP for OS/390 V3R2 defines four user exits. These exits return acceptance or reject codes. Depending on these codes, the FTP server will allow or deny the action that launched the user exit.

FTCHKIP This user exit is launched when a new connection is attempted on the OS/390 FTP server. The exit's parameters are the IP address of the client and the port number of the attempted connection. Use this exit to allow only a given set of IP addresses to access the OS/390 FTP server.

FTCHKPWD This user exit is launched when the user of an FTP client has entered his user ID and password. The exit's parameters are this user ID and password. Use this exit to enforce special access rules or extra password checking.

FTCHKCMD This user exit is launched each time the user of an FTP client has entered an FTP subcommand (such as GET, PUT, DIR, CD, and so on). The exit's parameters are the user ID, the subcommand, and the subcommand's arguments, if any. Use this exit to deny certain subcommands to some users.

FTCHKJES This user exit is launched each time an FTP client submits a job to the FTP JES. The exit's parameters are the user ID and the JCL buffer. The exit can accept the JCL, modify the JCL before returning an acceptance code, or just reject the JCL. Use this command to accept or reject specific JCL, possibly from only a given set of user IDs.

You might want to activate any or all of these user exits. Just write a program, define it as a given user exit, and it will be executed everytime an FTP client attempts the corresponding action during a connection to the OS/390 FTP server. For examples of FTP user exits, see Appendix B of *TCP/IP Version 3 Release 2 for MVS Implementation Guide*, SG24-3687.

11.4.2 Telnet and LUs

The Telnet component of TCP/IP for OS/390 supports a concept which is foreign to the regular UNIX world: logical unit (LU) mapping. An LU is, in SNA parlance, a device connected on a network. Each such device has a name called *LU name*, just like a TCP/IP host has a hostname. Here, an LU is equivalent to a terminal, such as a Telnet client window on a workstation.

Why bother with LU names? Well, LU names are to mainframe applications what IP addresses are to UNIX applications. If you have legacy applications that are designed to work with LU names, or if you have an SNA-type network management system, you need to give an LU name to every Telnet session. In other words, you need to map each Telnet session to an LU name.

This mapping is controlled in the PROFILE.TCPIP data set. By editing this file, you can:

- Map an IP address to an LU name. Each time you run a Telnet session from a workstation with a given IP, it will be mapped to the same LU.
- Map an IP address to an LU group. Using a Telnet emulator on a multi-windowed screen, you can open several Telnet sessions corresponding to only one IP address (that is, the address of the workstation). These Telnet sessions are mapped to an LU group, or group of LU names. The LU names in the LU group can be enumerated or listed as intervals in the PROFILE.TCPIP data set.
- Map an IP address set to an LU group. Using this setting, any Telnet session launched from workstations within a given set of IP addresses is automatically mapped to the first available LU name in a given group. The IP addresses composing the set can be either enumerated in the PROFILE.TCPIP data set or defined as a whole subnet.
- Define the startup application of the Telnet session. In the UNIX world, the user normally sees a generic login screen when the Telnet session is started. In OS/390, you can define the mainframe application that will be seen by the user when he starts the Telnet session. This *application mapping* is done on the basis of the IP address or group, the LU name or group, or the link name (which is the network interface name).

11.4.3 SMTP Without Sendmail

OS/390 UNIX System Services TCP/IP supports Simple Mail Transfer Protocol (SMTP) the electronic mail protocol of the Internet. However, instead of `sendmail`, it offers its own SMTP server, which is a started task.

The SMTP server can be configured to fulfill the following roles:

- As a gateway between Internet SMTP mail and NJE/RSCS sites (such as VM systems). The sample configuration file delivered on the installation media supports this function. It has to be customized for your site.
- As an SMTP mail server. It can receive SMTP mail and forward mail to other hosts.

The configuration data set of the SMTP server is SMTP.CONFIG. It roughly plays the role of sendmail in regular UNIX. The SMTP Server acquires the name of other hosts either by querying name servers declared in the TCPIP.DATA data set, or, if none is present, by looking in the local hostname tables.

11.5 Network Management

Many UNIX servers run applications that rely heavily on communications and exchange of information with other servers over the network. This section discusses some of the capabilities available in the OS/390 world.

11.5.1 Distributed Computing Environment (DCE)

OS/390 UNIX System Services offers Distributed Computing Services, a collection of services that help managing network resources as if they were all on the same machines.

11.5.1.1 Understanding DCE

The OS/390 UNIX System Services DCE Base Services (OSF DCE level 1.1) provide the strengths of a distributed computing environment, as follows:

- Transparency of data and logic
- Distributed, consistent directory service
- Security for both clients and servers integrated in execution path
- Scalability of distributed applications
- Interoperability and portability

OS/390 UNIX System Services DCE Services supports the following:

- Remote Procedure Call (RPC) lets calls between programs running on different platforms appear as local procedure calls to the programmer.
- Directory Services allows resources to be found anywhere in an enterprise without the need to know local names.
- Security Services solves security problems common in a distributed environment by handling identification and certification of users, clients, servers, and systems.
- Distributed Time Services synchronizes clocks running on different nodes.

All components supported are based on the Open Software Foundation (OSF) DCE level 1.1. The OS/390 UNIX System Services DCE Base Services support clients and servers that run on TCP/IP and SNA networks.

A common problem in server consolidation is the access to data distributed across the enterprise. Two DCE components are particularly helpful: Distributed File Service (DFS) and Network File System (NFS).

11.5.1.2 DCE Distributed File Service (DFS)

Distributed File Service is the file serving component of OSF DCE level 1.1.

OS/390 UNIX System Services DCE DFS for MVS/ESA permits users to access and share data in a distributed environment across a wide range of platforms, both IBM and non-IBM. It provides access to both OS/390 UNIX Services's Hierarchical File System and the OSF's DCE Local File System, thus helping managing an heterogeneous distributed environment.

OS/390 UNIX System Services DFS provides:

- Data consistency through advanced token management and client caching methods
- Uniform access to data through a naming convention that specifies unique file naming across enterprises
- Client support for record-level access to PS, PDS, PDS/E, and VSAM data sets
- Security based on Kerberos authentication
- High reliability and availability of data through replication across multiple servers with updates made automatically by DFS
- Improved manageability of distributed environments through the use of access control lists and distributed databases that track file location, authentication, and backup information

11.5.1.3 Network File System (NFS)

The Network File System allows an OS/390 UNIX System Services system to act as a file server to workstations, personal computers, or other authorized systems in a TCP/IP network. It also provides an MVS client. It enables client users to remotely access MVS data sets or OpenEdition MVS files from any system on a TCP/IP network that uses client software for the DFSMS/MVS Network File System protocol. The remote data sets or files are mounted from the mainframe to appear as local directories and files on the client system. DFSMS/MVS Network File System also provides access to the Hierarchical File System (HFS.)

Appendix A. Consolidation Candidate Selection

When server consolidation is being considered, it is frequently at a stage when no platform for the consolidation has been decided upon. Some financial or business benefit may be expected, but not quantified.

However, often nobody (customer or vendor) has a really clear understanding of the actual implementation effort, costs or the timetable involved.

The enterprise needs to assess and understand the following items very early in the decision making cycle:

Business requirements:

- Application availability
- Performance and I/O bandwidth
- Scalability
- Lower costs
  - Management
  - Operations
  - Capital
- Access to other data for electronic commerce or improved customer service
- Integration with other data or applications for electronic commerce or improved customer

Ease of migration

- Standard languages - for example C, C++, COBOL
- Standard databases - for example DB2, Oracle
- Standard networking - for example TCP/IP, TN3270, Motif, X-Windows
- Availability of the persons responsible for the design and/or coding of the application to be migrated
- Access to the vendor who supplied the current tools and application

The next section describes an IBM S/390 Division-developed methodology for assessing the suitability of an application for server consolidation.

A.1 Business Solution Assessment

Today, there is a need to understand numerous customer business and technical requirements, in order to implement new technologies in areas like Business Intelligence, ERP Applications, Server Consolidation, and Network Computing.

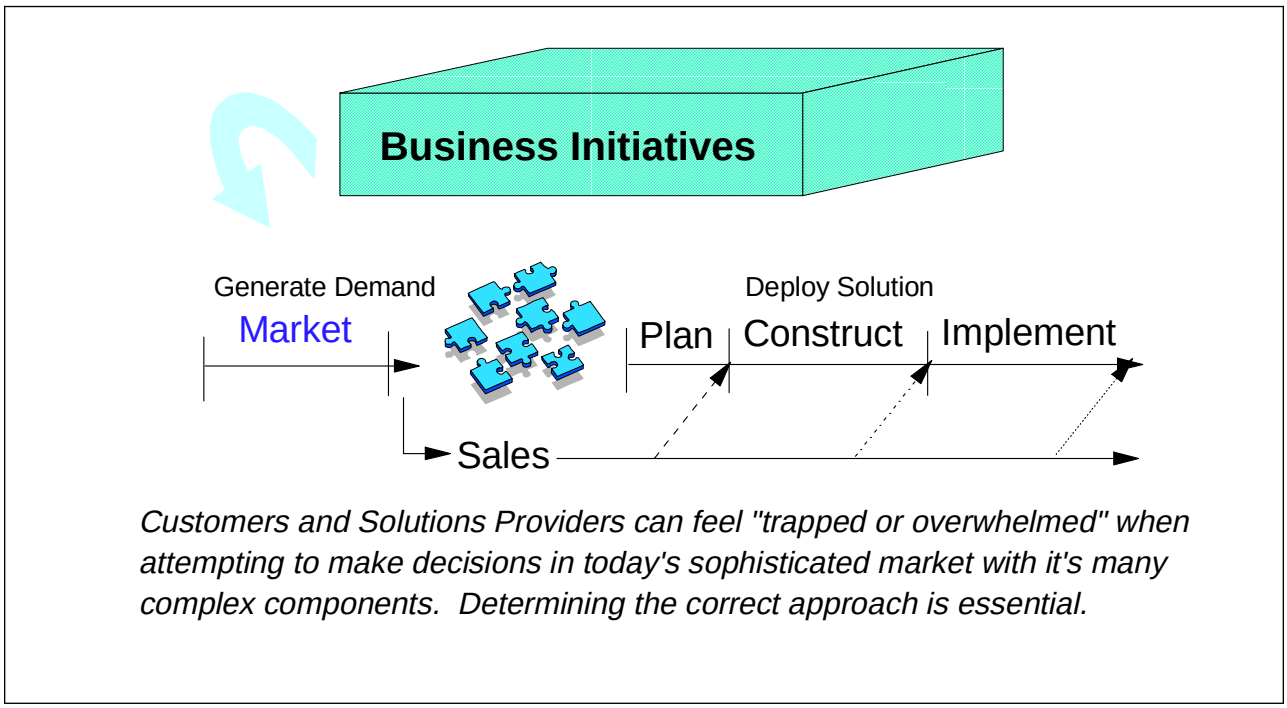


Figure 40. Today's Business Model

Having a clear understanding of these requirements enables you to have opportunities to move from a marketing activity into a S/390 solution. Factors that should be assessed in the early phases, and that will enable customers to move forward in their decision process as well as give IBM a means to provide the appropriate customer solution, include:

Customer:

- Budget

  What is the situation on costs? Everyone is concerned about costs, but is there or will there be a preset limit? Are all of the components clearly understood?

- Time

  What is the situation with regard to the project duration? Is there a requirement to be completed on a certain date? How will the date be decided?

- Frozen Period

  In the course of the effort, how will change management be handled? Can business function without enhancements for a period of time? How will the end user be affected?

- Resources

  Does the customer have the personnel to support the proposed effort? Will external support be required? Will an education plan be required?

- Availability

  What is the impact on the customer's end user, development and support environments?

- Performance

  What is required to support the target environment? Are there tools available? How will batch, online activity, database and network activity be monitored?

- Constraints

  Are there any inhibitors that should be identified, such as software that is not supported or which is outdated?

- Order of Importance

  What approach should be recommended? What offerings are available? In what sequence should a customer requirement be addressed?

- Third Party Involvement

  What Business Partners or ISVs are involved? What is the responsibility of third party companies?

- Costs

  What are the components for estimating costs?

- Resources

  What are the IBM and non-IBM resources required? Who are the correct IBM groups to contact?

For any server consolidation to S/390 there will be many participants. These will include Business Partners, ISVs, different parts of IBM's hardware, software and services offerings, and the Customer. There is a need to know how to identify and integrate these various components into the appropriate customer solution.

A.1.1 S/390 Business Solution Assessment - Proposed Business Model

The S/390 Business Solution Assessment is a planning program developed to assist in S/390 marketing and sales activities. This program will help optimize customer decisions by accurately understanding benefits, associated costs, risks and initial steps. This program is also an opportunity to ensure that all parties, the customer, IBM and third parties, have a clear understanding of the customer's business and technical requirements. This understanding will provide the information for a solution that will be based on our technology, products and expertise.

Figure 41 on page 180 shows an example of the proposed business model.

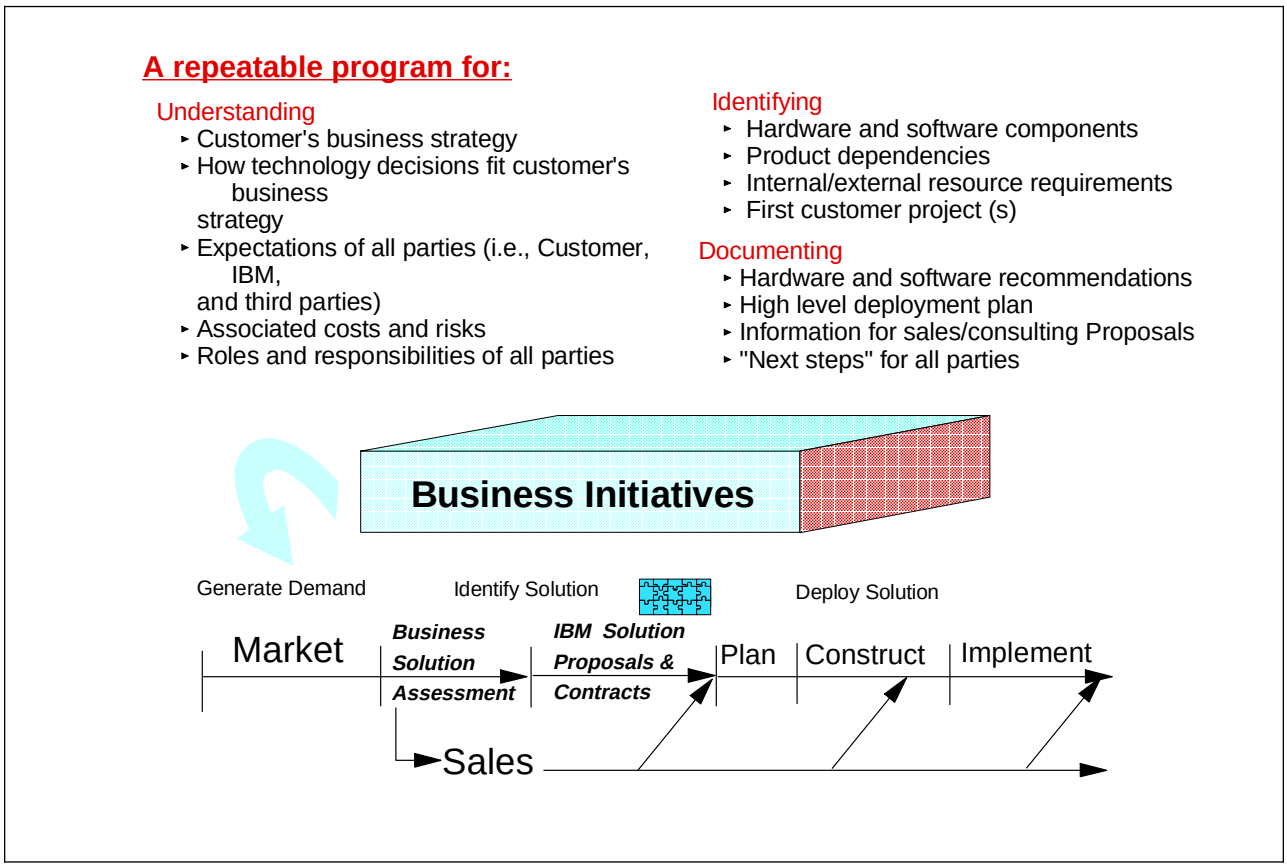


Figure 41. Proposed Business Model

Repeatable activities within this program provide a format used to understand, identify and document the information required to develop a S/390 server consolidation. Information gathered can be used to assist customers with decisions or considerations for environment preparation, hardware and software installation, selection of first end-user projects and internal/external skill requirements.

The initial steps of this program are focused on the things to do and put in place before beginning a S/390 server consolidation project. The initial steps are not really technical, although various levels of understanding of the technology is a base requirement. The initial steps must be to;

1. Reduce customer uncertainty by establishing credibility
2. Set the correct expectations for the customer, IBM and third party companies
3. Ensure that all the parties have a clear understanding of the expected business benefits and results
4. Ensure that the solution, and any technical decisions made, fits the customer's business strategy
5. Assist the customer with determining the proper next steps/phases

The goals of the S/390 Business Solution Assessment are to:

SET EXPECTATIONS related to the customer's business and technical requirements.

LEARN everything possible about the components that will effect the

customer decisions, such as complexity, size, maintainability, security, degree of conformance to standards, structure of strategic systems, change management, staffing, degree of end user satisfaction, database and application structure, and completion criteria.

EDUCATE the customer regarding the implementation of IBM and third party technologies that are available for specific S/390 server consolidations.

RECOMMEND alternatives based on what was learned during the assessment.

UNDERSTAND what IBM offerings, hardware, software, services, consulting and third party participation will be required for the proposed consolidation.

The S/390 Business Solution Assessment is an intense customer-site program, specifically designed to identify issues that customers will face when implementing a S/390 solution. This program will provide value for customers committed to or considering a S/390 server consolidation. The program consists of:

- Formal briefing and interview sessions with customer management, end user, and technical staff
- Third party discussions and presentations of their products and services
- Understanding of associated costs, benefits, and risks
- Evaluation of skills, education and implementation requirements
- Selection of candidates for first projects
- Recommendations for appropriate next steps

This program typically spans a one-to-three week period. The onsite portion of the assessment is typically two to five days of meetings and presentations with key customer staff and management. Data gathering, preparation, and some of the analysis will be done prior to the onsite activities. Hardware, software, services or consulting proposal preparation will occur during or after the onsite activities.

The Customer Profile form will provide a high level view of the customer's environment. This form will be used during an initial customer visit or conference call. During this visit, the following will occur:

- Discussions regarding concerns and issues
- A review of the S/390 Business Solution Assessment format
- Identification of the customer liaison for assessment support
- Presenting the customer with the S/390 Growth Initiative questionnaire
- Requests for information to analyze before the onsite session

The S/390 server consolidation questionnaire is used to quantify and qualify information regarding the customer's environment, hardware, software, applications, databases and so on. The questionnaire also provides the Engagement Manager with the information required to determine the assessment scope, agenda and involvement needed from various IBM groups and third party companies.

An agreement of the onsite schedule and agenda will be finalized with the customer. During this discussion, the customer, IBM and third party will agree on a start date and staff availability for the onsite S/390 Business Solution Assessment.

A.1.2 The S/390 Business Solution Assessment - Benefits

Success means meeting the customer's business and technical needs; otherwise, the real benefits for all parties may not be realized. The S/390 Business Solution Assessment is based on this concept. Assessments are structured as listening, studying and learning activities that can provide value to customers, IBM and third party for each S/390 Growth Initiative. The information gathered during the assessment is essential to ensure that the recommended solution will fulfill the real needs of the customer. There can be many components and alternatives for a given customer need. In order to provide value, a solution provider must ensure, early in the process, that the alternative selected and the recommended next steps are directly linked to what the customer considers as a business benefit.

In some cases, a customer's request for a server consolidation project can actually span other areas which can benefit from the strengths of the S/390 server, such as Network Computing, Business Intelligence, New Applications and so on. A solution provider many not be aware of this until some level of analysis or assessment is completed. Information gathered during an assessment may identify situations that may alter the original customer request. Factors may be discovered that could introduce unknown circumstances that should be considered part of the solution. These discoveries could provide the information required to understand where and how to begin.

IBM and third party solution providers will need to obtain the correct information during the early planning stages for a S/390 server consolidation to ensure that essential elements for a given customer solution are understood. There are clear benefits for all parties to take part in the S/390 Business Solution Assessment Program. Experience has shown us that all players must understand the overall objectives of the customer requirements and be clear on how to proceed. It may not be realistic to focus on only the technical aspects. In order to gain the benefits of today's market, solution providers must devise methods that associate hardware, software, services and partnerships to specific customer solutions.

For further details, customers should contact their local Client Representative.

Appendix B. Setting Up the Porting Project Environment

While porting your application to an OS/390 UNIX System Services environment, you can work in a familiar environment, using a UNIX shell that has the same look and feel and utilities as other UNIX environments.

If you want to be able to work with your workstation tools, and you have the Network File System feature on your workstation, you can mount a hierarchical file system (HFS) directory on your workstation's file system. Using NFS to mount an OS/390 UNIX System Services hierarchical file system to a workstation file system lets you use your workstation tools for tasks such as source code control, editing, and code quality checking.

This appendix on setting up the porting environment includes:

- Tuning the system
- Creating an HFS data set on the OS/390 UNIX System Services system
- Using the OS/390 UNIX System Services shell
- Using TCP/IP FTP to transfer archive files
- Data exchange and access
- Customizing the shell
- Compiling source code
- Checking your environment setup
- Finding tools and utilities

B.1 Tuning the System

Tuning the system is critical for performance. On the S/390 Web site `www.s390.ibm.com/products/oe` is a list of release-specific tuning targets, including a list specially designed for compile-intensive systems. Check with the OS/390 systems programmer to be sure your system has been tuned before you begin your porting project.

B.2 Creating an HFS Data Set on the OS/390 UNIX System Services System

The OS/390 systems programmer needs to set up an HFS data set mounted at /u/userid. Source files will then be stored in /u/userid, which is the home directory.

B.2.1 HFS Data Set

OS/390 UNIX System Services files are organized in a hierarchy. All files are members of a directory, and each directory is a member of another directory at a higher level in the hierarchy. The *highest* level of the hierarchy is the `root` directory.

OS/390 UNIX System Services views an entire file hierarchy as a collection of hierarchical file system data sets (HFS data sets). An HFS data set is a new type

of OS/390 data set for OS/390 UNIX System Services. Each HFS data set is a mountable file system.

A file in the hierarchical file system is called an HFS file. An HFS data set can contain one or many HFS directories (it will always have at least one) and zero or more files. The highest level directory of an HFS can be thought of as the root for that HFS, but be careful not to confuse this with the root of the entire file hierarchy. Note that when an HFS is initially created, its root will have permission 700 (which means that the file owner can read, write and execute, but all others have no access). You may want to broaden this access by changing it to 755, 775, or even 777 after mounting it the first time.

The Data Facility System-Managed Storage Hierarchical Storage Manager (DFSMShsm) provides automatic backup facilities for HFS data sets. ADSTAR Distributed Storage Manager (ADSM) provides backup function for HFS files. There are two types of backup:

- Incremental, in which all new or changed files are backed up
- Selective, in which the user backs up specific files

If DFSMShsm backs up an HFS data set that has not been used recently, it removes the data set from the current DASD volume. This frees up space of more active data sets. On the other hand, ADSM will perform the backup of files, but it does not free up the space for the file.

B.2.2 OS/390 DASD Space Allocation

In the S/390 world, disk space has traditionally been allocated in units of tracks or cylinders even if allocation in units of blocksize has been possible. The disadvantage of this method is the size of a track or cylinder often is different depending on the type of disk hardware.

More recently, it is possible to allocate space in terms of megabytes (MB 1024*1024 = 1048576 bytes) or kilobytes (KB 1024 bytes). Programmers from the UNIX world are more familiar with this method of allocating space and can better estimate how much space should be allocated in the HFS to hold their source code.

Table 13 is a short generalized table showing DASD allocation sizes for most commonly used DASD families.

Table 13. DASD Allocation

DASD type	Track size	Cylinder size	Tracks per Cylinder
3390	56 KB	850 KB	15
3380	47 KB	712 KB	15
9340	46 KB	697 KB	15

An HFS data set can have up to 123 secondary extents.

B.3 Using the OS/390 UNIX System Services Shell

Before you can move your data files to the OS/390 UNIX System Services system, you will need a user ID and password on the OS/390 system; the user ID must contain an OMVS RACF segment. The OS/390 systems programmer will set this up.

B.3.1 Accessing the Shell

After you have a user ID and password, you can access the OS/390 UNIX Services shell in one of these ways:

- With the OMVS command, from a logged on TSO/E user ID using a 3270 terminal or using a workstation running a 3270 emulator
- With the `rlogin` command, from a workstation running TCP/IP, or via the Outboard Communications Server (OCS)
- With the `telnet` command, from a workstation running TCP/IP, or via the Outboard Communications Server (OCS)

  Note: Be sure to check with the systems programmer for the system name and port address to be used for Telnet. There are two Telnet servers on OS/390 (one for TSO and one for the shell) and you need to get to the one for the shell. On many systems, the shell server is listening to port 623, instead of the well-known port 23 that all clients use by default.
- OCS login from a serially attached terminal

B.3.2 Editing

Three UNIX editors are supported:

- vi
- sed
- ed

You can use vi only from rlogin or Telnet sessions.

The ISPF editor is also available from the shell if you logged in via TSO/E; you cannot use the ISPF editor during an rlogin session. You invoke ISPF file edit using the oedit shell command or OEDIT TSO/E command; you can invoke file browsing using the obrowse shell command or OBROWSE TSO/E command.

Of course, if you use NFS to mount HFS files to your workstation, you can use your favorite workstation editor.

B.4 Using TCP/IP FTP to Transfer Archive Files

If TCP/IP is installed on both the workstation and the OS/390 system, you can use the File Transfer Protocol (FTP) facility to transfer your source data. If transferring single-byte data, FTP will convert the data from ASCII to EBCDIC for you. Binary data can also be sent using FTP if the binary parameter is specified.

Files are often bundled together in single files by utilities like pax, tar, and cpio. When these files are bundled into a single file, it is called an archive file. By convention, the file name of the archive indicates the utility that was used when the file was built. For example, a file named mvsport.tar indicates that the tar.utility

was used. The three utilities provide basically the same function: reading and writing of archive files. The important thing to know is that tar and cpio can only read and write files of their respective formats, but pax can read or write in either format. So given a pax, tar, or cpio file, you can use pax from the OS/390 UNIX System Services shell to “explode” or unwind the archive into its individual files.

To save disk space and transmission time, archive files can also be compressed by using the `-z` option with the pax, tar, and cpio utilities. The naming convention that is generally used for a compressed archive file is to end the filename with a .Z, for example, mvsport.tar.

With the TCP/IP OS/390 UNIX System Services Applications Feature installed on a remote OS/390 system, you can ftp files into or from that system's HFS.

Note: Be sure to check with the OS/390 systems programmer for the system name and port address you should use for ftp. There are two ftp servers on OS/390 (one for OS/390 datasets and one for HFS files). On many systems, the HFS server that comes with the application's feature is listening to port 621, instead of the well-known port 21 that all clients use by default.

An FTP client for the shell and utilities, ncftp, is available to download. Be sure to read the readme.MVS file before you unwind the tar file.

Note: If your browser does not work with the preceding link, use this URL instead:

`ftp://ftp.s390.ibm.com/u/ftp/os390/oe/port/ncftp.tar.Z`

With archive files, we recommend using the BINARY transfer option on FTP PUT and GET commands. If you do not use BINARY mode everywhere, your archive will most likely be corrupted and unusable.

B.5 Data Exchange and Access

The implementation of the HFS is transparent to the type of data to be stored. Data may be text or binary; the implementation does not care about this. However, you may need to know what kind of data you are working with when you exchange data with other systems:

- Text Data

  In a client/server environment where OS/390 participates, you have to consider how to do ASCII-EBCDIC translation. At first this might seem to be only a matter of two translation tables, but it goes much further than that. As soon as you have to deal with different countries, you have to handle translations using the appropriate code page. OS/390 UNIX System Services provides:

  – The `iconv` command in the shell environment
  – The iconv() routine for programs coded in C

  Both allow you to convert according to the code pages that are applicable.

- Binary Data

  Binary data is much more complicated to handle than text data. Integer variables may be stored differently according to the byte order in storage (little endian (PC) versus big endian (S/390)). Different representations of floating point variables exist. There is no single simple solution for handling such differences. Each situation has to be analyzed to find a suitable solution.

In an environment where S/390 provides the server platform, you normally can make a trade-off based on what kind of coding is mostly needed. If workstation users store and fetch data in the HFS based on NFS or FTP and further processing on the OS/390 system is limited, then you may decide to store all data in the representation it has on the workstations.

To transfer files in and out of your OS/390 UNIX System Services system, you can use the FTPD server from the TCP/IP OS/390 UNIX System Services Applications Feature. With this server, you can transfer files directly in and out of the HFS, as well as in and out of traditional OS/390 data sets. You can use the FTPD server for transferring both binary data and text data (C source code). You can transfer single files, tar files, cpio files, and pax files. The latter are collections of files which make it easier to transfer a bunch of files and even allow you to reconstruct the directory structure. Use tar, cpio, or pax files when exchanging data which span several directories. The drawback is that you have to separate text and binary data.

As we mentioned before, file archives of tar, cpio, or pax format are very convenient to transfer files that are organized in a directory tree. The `pax` shell command allows you to expand the various kinds of archive file formats (cpio or tar). It allows you to translate data from ASCII to EBCDIC or vice versa at the same time. Here is an example of how to extract a tar type archive and translate from ASCII-to-EBCDIC at the same time:

```
pax -o from=ISO8859-1,to=IBM-1047 -rf your_tar_file.tar
```

Data exchange between traditional OS/390 data sets and the HFS can be accomplished through the TSO/E copy commands that are provided with OS/390 UNIX System Services: OCOPY, OPUT, OGET, OPUTX and OGETX.

- OCOPY is based on DDNAME allocation. So you have to allocate your input and output DDNAMEs to OS/390 data sets or HFS files before you invoke the OCOPY command.
- OGET, OPUT, OGETX and OPUTX give you the opportunity to specify the OS/390 data set name and HFS file name directly on the command invocation.
- OGETX and OPUTX support partitioned data sets, allowing you to copy all members of a partitioned data set in one command invocation. In addition you may apply a suffix to the new files. If, for example, you want to copy all members of your partitioned library hlq.project.c to the HFS as files in a specific directory, you can have all member names changed to file names of the form membername.c while you copy.

All commands support ASCII-to-EBCDIC conversion.

When you copy from a traditional OS/390 data set that contains binary data, you should use the BINARY option of the OPUT or OPUTX command:

```
OPUT GZIP.EXE '/u/hdm/gzip' BINARY
```

In case you copy a data set that contains text type data in ASCII encoding and you want to convert the data to EBCDIC while copying to the HFS, use this command format:

```
oput love.letter '/u/hdm/love.letter' binary CONVERT((BPXFX111))
```

Note: You need both parentheses for the CONVERT keyword parameter. The BINARY option is important, because it tells OPUT to handle the input data set as a string of bytes, not as a collection of records.

B.6 Customizing the Shell

If you are using the OS/390 UNIX System Services shell for the first time, you might need to customize the following areas before you start to compile and link your code:

- Environment variables
- Square brackets
- Using make
- Header files
- Libraries for functions and headers

B.6.1 Environment Variables

You may want to customize environment variables associated with the `c89/cc/c++` utility and environment variables in your $HOME/.profile. The `c89/cc/c++` command is the interface to the OS/390 C/C++ compiler, the prelinker, and the linkage editor for OS/390 UNIX System Services. The `c89/cc/c++` command can be invoked directly from the shell or a batch job. For improved shell performance, there are two shell variables to customize: `_BPX_SHAREAS` and `_BPX_SPAWN_SCRIPT`.

For more information on customizing environment variables, see *OS/390 OpenEdition Planning* and *OS/390 OpenEdition User's Guide*. For more information on c89, cc and c++, see *OS/390 OpenEdition Command Reference*.

B.6.2 Using Square Brackets

When you use OMVS functions from 3270 terminals, you will encounter problems with the square brackets ([ ]). This is true even in a native US environment that uses host code page 037. To be able to enter, display, and print square brackets correctly, you need to customize the keyboard mapping on your workstation, and also ensure that OS/390 UNIX System Services and any 3270 emulation package that you might be using are both using the same code page.

Here is one solution, which applies to Communications Manager/2 Version 1.11:

- To find out which host code page you are using, click **Keyboard** in an active 3270 window. Select **Remap keyboard** from the pulldown menu. The remapping utility starts. Click **File** on the menu bar and select **Properties** Now you should see a panel that includes information about the active host code page.
- CM/2 provides a tool to extract all code page tables from its internal database. This tool is called ACSGCCRT.EXE and it creates *.cpt files. Switch to \cmlib and run this tool.
- Assuming you are using host code page 037, make a backup copy of 037.cpt.
- Now edit 037.cpt and apply the following changes (see Figure 42 on page 189 for the original version of 037.cpt, and Figure 43 on page 189 for the changed version of 037.cpt).

1. Change the lines that assign Left_Bracket to BA and Right_Bracket to BB to Undefined for BA and BB.
2. Then change the Y_Acute_Capital (assigned to AD) to Left_Bracket.
3. Change Diaeresis/Umlaut_Accent (assigned to BD) to Right_Bracket.

- In addition to this, it may be necessary to remap your keyboard so the keys that have the square brackets engraved are really assigned to square bracket characters. Go to the keyboard remap previously described, click the corresponding keys, and change both if necessary.

```
Y_Acute_Capital                 AD
Thorn_Icelandic_Capital         AE
Registered_Trademark_Symbol     AF
Circumflex_Accent               B0
Pound_Sterling_Sign             B1
Yen_Sign                        B2
Middle_Dot                      B3
Copyright_Symbol                B4
Section_Symbol                  B5
Paragraph_Symbol                B6
One_Quarter                     B7
One_Half                        B8
Three_Quarters                  B9
Left_Bracket                    BA
Right_Bracket                   BB
Overline                        BC
Diaeresis/Umlaut_Accent         BD
```

Figure 42. Original Version of 037 as Provided with CM/2

```
Left_Bracket                    AD
Thorn_Icelandic_Capital         AE
Registered_Trademark_Symbol     AF
Circumflex_Accent               B0
Pound_Sterling_Sign             B1
Yen_Sign                        B2
Middle_Dot                      B3
Copyright_Symbol                B4
Section_Symbol                  B5
Paragraph_Symbol                B6
One_Quarter                     B7
One_Half                        B8
Three_Quarters                  B9
Undefined                       BA
Undefined                       BB
Overline                        BC
Right_Bracket                   BD
```

Figure 43. Modified Version of 037

For other code page tables, the procedure described here may also apply. But additional problems may occur with the other POSIX invariant characters listed in Figure 44 on page 190.

```
Right brace              }
Left brace               {
Backslash                \
Right square bracket     ]
Left square bracket      [
Circumflex               ^
Tilde                    ~
Exclamation point        !
Pound sign               #
Vertical bar             |
Dollar sign              $
Commercial at-sign       @
Accent grave             `
```

Figure 44. POSIX Invariant Characters

Assuming your host code page is 273 (German), you will have duplicate assignments of code points. So as you can see, a general solution is not that simple.

Each 3270 emulator is likely to require slightly different customization. For the ISPF edit environment, you can select ISPF option 0 and then set the terminal type to 3278A.

It should be pointed out that there is no general recommendation. A different approach may be to customize an OS/390 UNIX System Services user conversion table, BPXFXnnn. However, in this case you have a solution only in the OMVS shell environment, not in the ISPF edit environment.

B.6.3 Using make

Larger applications will probably have a collection of makefiles that are used to build the object files and executables from source files.

The make utility is a recipe-driven utility for managing the compilation process. The programmer specifies relationships between programs; make ensures that the compiles, links, and so on are done correctly and at the right time. The make command calls the c89/cc/c++ utility by default to compile and link programs specified in your makefiles. The make implementation varies across UNIX platforms. OS/390 make tends to be less tolerant of non-standard files than other makes. With each new release of OS/390, there are fewer differences between OS/390 make and other makes. See *OS/390 OpenEdition Programming Tools* for more information on the `make` command, and *OS/390 OpenEdition Command Reference* for the syntax of the `make` command.

B.6.3.1 .POSIX makefile Special Target

In your makefile, specify the .POSIX special target (notice the dot “.” at the beginning of the target name). This causes make to process the makefile as specified in the POSIX.2 standard.

Note: This special target must appear `before` the first noncomment line in the makefile. Do not associate any prerequisites or recipes with this target.

The .POSIX target:

- Causes make to use the shell when running all recipe lines (one per shell).

- Disables any brace expansion (set with the .BRACEEXPAND special target).
- Disables metarule inferencing.
- Disables conditionals.
- Disables make's use of dynamic prerequisites.
- Disables make's use of group recipes.

B.6.3.2 gnumake

A ported version of gnumake is available on the `www.s390.ibm.com/products/oe` Web site. gnumake executes commands in a makefile to update one or more target names, where name is typically a program.

B.6.4 Header Files

There are a couple of environment variables that are used to point the c89/cc/c++ utility to the header files. For example:

- Environment variable `{_INCDIRS}` is set, by default, to search the /usr/include directory.
- Environment variable `{_CLIB_PREFIX}` is set, by default, to search the OS/390 data sets that contain the C/C++ compiler header files and the compiler messages.

The *OS/390 OpenEdition Command Reference* has more information about the environment variables that affect the c89/cc/c++ utility.

B.6.4.1 Socket Header Files

To get access to the OS/390 UNIX System Services socket header files when compiling, consider using the following options when you invoke the c89/cc/c++ utility:

-DMVS

This enables logic in include files that is unique to OS/390 (such as referencing variables defined in the DLL).

_OE_SOCKETS

This defines a BSD-like socket interface for the function prototypes and structures involved. This can be used with `_XOPEN_SOURCE_EXTENDED 1` and the XPG4.2 socket interfaces will be replaced with the BSD-like interfaces. An application cannot specify `_OE_SOCKETS` with `_OPEN_SOCKETS`. Otherwise, a compile time error message will be generated.

_OPEN_SOCKETS

This defines a BSD-like socket interface for the OS/390 UNIX System Services Application Services (FMID HOT11x0). This option cannot be used with either `_XOPEN_SOURCE_EXTENDED 1`, `_OE_SOCKETS`, or `_OPEN_SOURCE=2`. A compile time error message will be generated.

Use of this feature test macro implies that the application is using the OS/390 UNIX System Services Application Services (FMID HOT11x0) product implementation of the sockets runtime library, and must comply with several other requirements, such as header concatenation and inclusion of HOT11x0 unique headers.

_ALL_SOURCE

This defines all of the functionality that is currently available on OS/390 UNIX System Services, including XPG4, XPG4.2, and all of the OS/390 UNIX System Services extensions. In addition, defining `_ALL_SOURCE` makes a number of symbols visible that are not permitted under ANSI, POSIX or XPG4, but which are provided as an aid to porting C language applications to OS/390 UNIX System Services.

Note: If a source program can be ported to OS/390 UNIX System Services just by defining `_ALL_SOURCE`, then it is possible to set this option on the command line invocation of the compiler:

```
c89 -D _ALL_SOURCE ....
```

If you do not specify any option, you will be using Universal UNIX (UU) sockets that are compatible with C++.

B.6.5 Libraries for Functions and Headers

LE Runtime Library functions are kept in OS/390 data sets rather than an HFS archive library like /lib/libc.a. In OS/390 UNIX Services, the default directory to search for archive libraries set by the `{LIBDIRS}` environment variable is /lib followed by the /usr/lib directory. This default is set to be consistent with other UNIX implementations, but the library functions are not contained in those directories. Instead, the OS/390 data sets that are installed with Language Environment (LE) are used to resolve library functions. For more information on which LE data sets are searched, see the description of `{_PLIB_PREFIX}` in *OS/390 OpenEdition Command Reference*.

Functions used by a program should be declared in the program. This is true whether you create them or you use system calls provided by the platform. If you do not declare the functions, the C compiler will create default declarations that may or may not operate correctly.

System calls do not need to be declared explicitly, as their declarations are part of the header and include libraries that you specify to gain access to system calls. These libraries are generally installed in the directory /usr/include in files with a .h suffix on most systems. OS/390 UNIX System Services header files are installed in OS/390 data sets, but generally they are also installed in the HFS and you can grep them. There are a few exceptions, such as headers for C++ classes, which are not installed in the HFS.

B.7 Compiling Source Code

After OS/390 UNIX System Services has been installed, ask the OS/390 systems programmer to run the setup checker, which can be downloaded from the OpenEdition Web site `www.s390.ibm.com/products/oe`. Among other things, the setup checker verifies that you can compile and run a program.

If you have loaded the source code into the HFS, customized the c89/cc/c++ utility for your shell session, and taken a look at your makefiles and made any necessary changes, it is time to start compiling your code. You can run your code through a code checker or lint filter that uses the OS/390 UNIX System Services header files to possibly flag any nonconforming and unsupported constructs.

B.7.1 Default Settings

If you run with the c89/cc/c++ default settings, the following are true:

- C source file names end with the .c suffix
- C++ source file names end with the .C suffix
- Archive file names end with the .a suffix
- The `-c` option of the `c89/cc/c++` command specifies that only compilations will be done.
- Unless you name the executable file with a `-o filename` option, the default name is a.out.

These topics which are discussed in the following sections, explain how the compiler works in the OS/390 UNIX + Services environment:

- `-o` option at the end of the `c89/cc/c++` command
- Compiler options
- Conditional compilation

Another resource for compiler tips is the Compiler questions and answers Web page.

B.7.2 -o option at the End of the c89/cc/c++ Command

In accordance with the POSIX.2 and XPG4 standards, options and operands of utilities cannot be mixed: all options must appear `before` operands. You cannot put the `-o` option at the end of the command. The X/Open compliance suites specifically test to ensure that c89 does not allow the mixing of operands and options.

Because the mixture of operands and options is a common practice on UNIX platforms, OS/390 UNIX System Services lets you enable this with an environment variable (a different one for each utility). If you typically use this syntax or use makefiles that imply this practice, you may want to add one or more of these variables to your $HOME/.profile file:

```
export _C89_CCMODE=1
export _CXX_CCMODE=1
export _CC_CCMODE=1
```

B.7.3 Compiler Options

LANGLVL(EXTENDED)

The C library contains several functions that are extensions to the SAA CPI Level 2 definition. These library functions are available only if the `LANGLVL(EXTENDED)` compile-time option is in effect. Some of the stub routines for the extensions are available if you specify `LANGLVL(ANSI)`. They are made available for compatibility with Version 1; they may not be available in the future. (Within runtime libraries, a `stub` routine is a routine that contains the minimum lines of code required to locate a given routine at run time.)

The `LANGLVL(EXTENDED)` option can also make the compiler more “forgiving” allow you to compile code that it ordinarily would complain about (type mismatches and so no).

To specify this option to c89/cc/c++, use:

`-W0,langlvl(extended)`

However, you should take care to protect the parentheses from shell interpretation. One technique to avoid this problem is to specify this option implicitly for all compiles (either in your profile or from the command line):

```
export _CC_OPTIONS='-W0,langlvl(extended)'
```

-g option for compile/linkedit
Use this option for enhanced diagnostic messages.

-O or -2 for compile/linkedit
For automatic inlining, use either the `-O` or `-2` option. This is recommended for any performance-sensitive final code.

-V option for compile/linkedit
For listings, use the `-v` option.

-D_OPEN_SYS
Use this option to indicate that symbols required by POSIX.1, POSIX.1a, POSIX.2 are made visible. Any symbols defined by the `_OPEN_THREADS` macro are also made visible. Additional symbols can be made visible if any of these standards explicitly allows the symbol to appear in the header in question or if the symbol is defined to be an OS/390 UNIX System Services services extension.

For more information about the feature test macros, see *OS/390 C/C++ Library Reference*.

B.7.4 Conditional Compilation

One of the problems programmers have is writing code that can work on many different machines. In theory, C code is portable; in reality, many machines have small differences that must be accounted for. The compiler allows the programmer great flexibility in changing the way code is generated through the use of conditional compilation. If you find that a function works differently or a header file declaration is different under OS/390 UNIX System Services, you can insert `#ifdef` and `#endif` statements in the code:

```
#ifdef os390
#include <stdlib.h>
#else
#include <malloc.h>
#endif
```

To include all the code in between the `#ifdef os390` and the `#endif` statements, use the `-Dos390` compiler option (which defines the os390 macro) in the `c89/cc/c++` command invocation or in your makefile.

B.8 Checking your Environment Setup

After you have set up your OS/390 UNIX System Services environment, you can test it with a simple port, such as gzip. Try the instructions for porting gzip to OS/390 UNIX System Services.

B.9 Finding Tools and Utilities

Each development group has its own standards and favorites among the many tools and utilities widely used in the UNIX environment. Some are available on OS/390 UNIX System Services, others are easily ported.

A number of ISVs have ported gnumake to OS/390 UNIX System Services as part of their porting projects; gnumake is now available on our tools and toys page (`http://www.s390.ibm/products/oe/bpxa1toy.html`). On this page, there are a number of tools and toys you can download, and the list keeps growing.

Appendix C. Ported Tools

The ported packages provided here worked to the satisfaction of the people who ported them.

A fuller list may be found on:
`http://www.s390.ibm.com/products/oe/bpxa1toy.html`

Note: If you prefer, you can do an anonymous login to the S/390 FTP server to get a file, as follows:

```
ftp www.s390.ibm.com
(enter "anonymous" for user name)
cd os390/oe/port   (or:  cd os390/oe/toys)
dir                (to list the files and subdirectories)
get README.txt
get HOW-TO.txt
binary
get xxxxxxx.xxx.Z   (the name of the package you want)
```

C.1 Freeware and Public Domain Packages

Package	*Description*
ACE	ACE is an object-oriented network programming toolkit in C++. Contact: `gehr@sweng.stortek.com`
banner	banner prints out text in large block letters. Contact: `pfuntner@vnet.ibm.com`
cpost	cpost lets you “pretty print” C source for easier analysis. You run cpost against one or more C source and header files, and it generates a postscript file for printing. Contact: `codeczuk@us.ibm.com`
dTET2	X/Open's Distributed Test Environment Toolkit Version 2. Contact: `atotten@vnet.ibm.com`
gnumake	gnumake is used to maintain groups of programs. It executes commands in a makefile to update one or more target names, where *name* is typically a program. Contact: `josh@watson.ibm.com`
gzip	A popular tool for compressing data in files, gzip is similar to the compress utility but more effective. In addition, gzip is smart enough to handle *.Z and *.z files. Contact: `pfuntner@vnet.ibm.com`
ksh93	A revision of the ksh88 shell, ksh93 offers associative arrays, floating-point arithmetic, and POSIX conformance, yet is backwards-compatible with ksh88. After you go to `http://www.research.att.com/sw/tools/reuse`, follow these steps: • Under Non-Commercial License Agreement and Available Software, click **binary**. • Select **ast-base-97** and look for UNIX: IBM OS/390 UNIX System Services.

Contact: dgk@research.att.com

m4 The m4 macro processor is a front-end processor for a compiled (or assembled) programming language, such as C or C++.
Contact: swehr@us.ibm.com

ncftp ncftp is an FTP client for the shell. Be sure to read the README.mvs file after you unwind the tar file.
Contact: pfuntner@vnet.ibm.com

perl perl is a language for easily manipulating text, files, and processes.
Contact: pfuntner@vnet.ibm.com

ping ping is a socket program used by network administrators to diagnose network problems. It provides feedback on the reliability of the connection and the time required to reach the target host.
Contact: trawick@ibm.net

Samba Samba is a suite of programs which work together to allow a client to access a server's filespace and printers via the SMB (Server Message Block) protocol.

uuencode and uudecode
uuencode is useful for packaging binary files prior to mailing them to other systems, while uudecode is used to read a uuencoded file and recreate the original.
Contact: pfuntner@vnet.ibm.com

wall wall is a utility that a superuser can use to broadcast a message to all shell users who are logged on.
Contact: swehr@us.ibm.com

C.2 OS/390 UNIX System Services Tools

These tools were designed for OS/390 UNIX System Services by OS/390 UNIX System Services developers and testers. Unless otherwise noted, most of the tools are not very portable. They probably will not work well on other UNIX systems without at least a little modification.

Package ***Description***

bpxwdyn bpxwdyn is a dynamic allocation and dynamic output interface for REXX or C. The code for bpxwdyn was integrated into OS/390 UNIX Services in MVS 5.2.2. This is a link to the documentation for the interface. (Be sure to download and read the README file too.)
Contact: schoen@vnet.ibm.com

cploadmod cploadmod is a shell command that, using the binder, copies load modules between PDS load libraries and HFS files. (Be sure to download and read the README file too.)
Contact: schoen@vnet.ibm.com

dirsize dirsize displays the amount of data contained in a directory and all its subdirectories (similar to the UNIX `du` command). This tool should be very portable to other POSIX.1 platforms. It uses no functions exclusive to OS/390 UNIX Services.
Contact: pfuntner@vnet.ibm.com

getuids getuids is a utility that displays information about OS/390 UNIX Services users and groups from the security database, similar to what would be found in the /etc/passwd file on other UNIX systems. (Be sure to download and read the README file too.)
Contact: marcw@vnet.ibm.com

ifind ifind searches a file system for files that share data with other files ("show me files that are links to file abc") or, if given an inode, searches for a file.
Contact: pfuntner@vnet.ibm.com

libascii The C/C++ run-time library functions support EBCDIC characters. The libascii package provides the C/C++ run-time library functions an ASCII interface layer for some of the more commonly used C/C++ run-time library functions.
Contact: libascii@vnet.ibm.com

mcp mcp is a shell function wrapper for readmvs and writemvs (both are described in the following sections), and all three are packaged together as readmvs. mcp mimics the syntax of cp for copying files from one place to another. These files may be OS/390 files, indicated by a leading // on each OS/390 file name. "OS/390 file" means a sequential data set or a member of a partitioned data set. (Be sure to download the the README file too.)
Contact: marcw@vnet.ibm.com

oesvp The OS/390 UNIX System Services Setup Verification Program (SVP) lets you check for troublesome setup errors before they trip you up. This utility requires ISPF version 4.1 or higher and MVS 5.2.2 or any release of OS/390. (Be sure to download the README file too.)
Contact: swehr@us.ibm.com

omvstape omvstape is actually two utilities that allow you to read and write from and to tapes in an OS/390 UNIX System Services shell session.
Contact: pfuntner@vnet.ibm.com

osendmail osendmail sends mail to remote users from the OS/390 UNIX System Services shell.
Contact: pfuntner@vnet.ibm.com

passwd The passwd utility changes the login password for the user ID specified. (Be sure to download the README file too.)
Contact: marcw@vnet.ibm.com

perr Given an error code or reason code in hexadecimal, the perr utility outputs the message text. (Be sure to download the README file too.)
Contact: schoen@vnet.ibm.com

pschart pschart displays processes in a way that emphasizes the parent/child relationships between the processes.
Contact: pfuntner@vnet.ibm.com

qftp qftp is a shell command that uses the ftp utility distributed with TCP/IP to give a shell user some ftp capabilities. (Be sure to download the README file too.)
Contact: schoen@vnet.ibm.com

readmvs and writemvs
: readmvs is a utility that copies one OS/390 file to stdout, so it can be manipulated in shell pipelines and redirected into HFS files. writemvs is a utility that copies stdin to an OS/390 file. ("OS/390 file" means a sequential data set or a member of a partitioned data set.) readmvs, writemvs, and mcp are packaged together. (Be sure to download the README file too.)
Contact: `marcw@vnet.ibm.com`

rexxfunc
: rexxfunc is a REXX package for the REXX I/O functions and several other useful functions. (Be sure to download the README file too.)
Contact: `schoen@vnet.ibm.com`

shcmd
: shcmd is a REXX external function that runs a shell command, and traps output from the command or pipes input to the command. (Be sure to download the README file too.)
Contact: `schoen@vnet.ibm.com`

stat
: stat displays `stat()` information about specified files, like a verbose ls command.
Contact: `pfuntner@vnet.ibm.com`

startd
: Use the startd utility to start one or more daemons that operate synchronously (that is, they receive control and do not return control immediately back to the invoker.) The startd utility also ensures that the started daemons are protected from unexpected signals such as SIGHUP, SIGINT, SIGQUIT, and so on. (Be sure to download the README file too.)
Contact: `marcw@vnet.ibm.com`

submit
: submit is a shell command that submits JCL. (Be sure to download the README file.)
Contact: `schoen@vnet.ibm.com`

tsocmd
: tsocmd is a shell command that creates a full batch Terminal Monitor Program (instead of using your TSO session) and runs a TSO command. This command supports authorized TSO commands, which the tso shell command does not. (Be sure to download the README file.)
Contact: `schoen@vnet.ibm.com`

uurestore
: uurestore. restores trailing blanks in a uuencoded file.
Contact: `pfuntner@vnet.ibm.com`

C.3 More Information on Some Packages

This section contains more information about some of the previously mentioned packages.

C.3.1 ACE

ACE is a freely available object-oriented toolkit that provides a set of reusable C++ wrappers and higher level class categories and frameworks that provide common communication software tasks across a wide range of operating systems, now including OS/390 UNIX System Services. All of the changes that were necessary for the OS/390 UNIX System Services port have been incorporated in the latest ACE release.

The communication software tasks provided by ACE include event demultiplexing and event handler dispatching, service initialization, interprocess communication, shared memory management, message routing, dynamic (re)configuration of distributed services, multithreading, and concurrency control.

C.3.2 m4

The m4 macro processor is a front-end processor for a compiled (or assembled) programming language. The `#define` statement in C language is an example of the facility provided by the macro processor. At the beginning of a program, you can define a symbolic name or symbolic constant as a particular string of characters. The m4 macro processor then replaces later unquoted occurrences of the symbolic name with the corresponding string. Besides replacing one string of text with another, the m4 macro processor provides the following features:

- Arithmetic capabilities
- File manipulation
- Conditional macro expansion
- String and substring functions

C.3.3 perl

The text processing language perl provides a more concise and readable way to do many jobs that were formerly accomplished (with difficulty) by programming in the C language or one of the shells. Even though perl is not yet a standard part of UNIX, it is likely to be available at many UNIX sites. The archive file is quite large (even though it is compressed).

Be sure to read ./README.first after you unwind the archive for some notes about the port and special things that were included. This version is ported from code developed by Larry Wall. Read both the GNU General Public License and the Artistic license that is included with this code.

C.3.4 uuencode and uudecode

The ported uuencode and uudecode include ASCII/EBCDIC conversions so that uuencode can create files that any uudecode utility can use and uudecode can use an encoded file that any uuencode utility produces.

With MVS 5.2.2, uuencode and uudecode are packaged with the OS/390 UNIX System Services Shell and Utilities. These ported versions are only necessary if your OS/390 UNIX System Services system is earlier than OS/390 and MVS 5.2.2. They are not intended to be used on systems without OS/390 UNIX System Services.

Before running uudecode on a file that might have had trailing blanks removed, you might want to run uurestore on the uuencoded file.

C.3.5 omvstape

Since our cpio, tar, and pax utilities do not have tape support, the omvstape tools can be used to manage tapes. They will work equally well with any utilities that read a lot of data from stdin or write a lot of data to stdout.

C.3.6 osendmail

osendmail uses the SMTP server, much like the TCP/IP SMTPNOTE command in TSO/E does. Following is an example of its usage:

```
echo "It works!" | osendmail pfuntner@vnet.ibm.com
```

C.3.7 uurestore

IBM mailservers have a habit of removing trailing blanks, altering the contents of the resulting file. Before running uudecode on a file that either has or could have had trailing blanks removed, run uurestore on the uuencoded file. This program should be very portable.

Appendix D. Special Notices

This publication is intended to help customers who are considering the consolidation of multiple UNIX servers and where an IBM S/390 may be under consideration for such a project. The information in this publication is not intended as the specification of any programming interfaces that are provided by OS/390 Versions 1 or 2, or of any other IBM subsystem that operates under OS/390.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AD/Cycle®	ADSTAR®
AIX®	AS/400®
BookManager®	C/MVS
C/370	CICS®
DB2®	DFSMS
DFSMS/MVS®	DFSMShsm

DRDA®
ESCON®
IMS
Magstar
MQSeries
MVS/ESA
NetView®
OpenEdition®
OS/390
POWERparallel®
Predictive Failure Analysis®
PS/2®
RAMAC
RMF
S/390®
SAA®
SP
ES/9000®
IBM®
LoadLeveler®
Micro Channel®
MVS® (logo)
NetFinity®
OPC
OS/2®
Parallel Sysplex
PR/SM
PROFS®
RACF
Resource Measurement Facility
RS/6000
S/390 Parallel Enterprise Server
Scalable POWERparallel Systems®
400®

The following terms are trademarks of other companies:

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be trademarks or service marks of others.

Appendix E. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

E.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 207.

- *Selecting a Server - The Value of S/390*, SG24-4812
- *DB2 for OS/390 Performance Topics*, SG24-2213
- *OS/390 Security Server Audit Tool and Report Application*, SG24-4820
- *TCP/IP Version 3 Release 2 for MVS Implementation Guide*, SG24-3687
- *Porting Applications to the OpenEdition MVS Platform*, GG24-4473
- *Accessing OpenEdition MVS from the Internet*, SG24-4721
- *OS/390 Security Server Audit Tool and Report Application*, SG24-4820

E.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

E.3 Other Publications

These publications are also relevant as further information sources:

- *OS/390 OpenEdition Programming Tools*, SC28-1904
- *OS/390 OpenEdition Command Reference*, SC28-1892
- *OS/390 OpenEdition Planning*, SC28-1890
- *OS/390 OpenEdition Users Guide*, SC28-1891
- *OS/390 V1R3 C/C++ Language Reference*, SC09-2360
- *OS/390 C/C++ Run-Time Library Reference*, SC28-1663
- *OS/390 Security Server (RACF) Command Language Reference*, SC28-1919

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at `http://www.redbooks.ibm.com/`.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type `GOPHER.WTSCPOK.ITSO.IBM.COM`
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type the following command:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
  ```

  For a list of product area specialists in the ITSO: type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Web Site on the World Wide Web**

  `http://w3.itso.ibm.com/redbooks/`

- **IBM Direct Publications Catalog on the World Wide Web**

  `http://www.elink.ibmlink.ibm.com/pbl/pbl`

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to `announce@webster.ibmlink.ibm.com` with the keyword `subscribe` in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1)001-408-256-5422 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to `softwareshop@vnet.ibm.com`
- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com/
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to `announce@webster.ibmlink.ibm.com` with the keyword `subscribe` in the body of the note (leave the subject line blank).

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

□ Invoice to customer number

□ Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Index

R

S

T

U

V

W

ITSO Redbook Evaluation

Consolidating UNIX Systems onto OS/390
SG24-2090-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.ibm.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

SG24-2090-00
Printed in the U.S.A.