IBM Advanced Interactive Executive/370
(AIX/370)
Planning Guide
Version 1.2.1

Document Number GC23-2065-01

IBM Advanced Interactive Executive/370
(AIX/370)

Planning Guide

Version 1.2.1

Document Number GC23-2065-01

Edition Notice

Second Edition (March 1991)

This edition applies to Version 1.2.1 of the IBM Advanced Interactive Executive for the System/370 (AIX/370), Program Number 5713-AFL, and to all subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for reader's comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Corporation, Department 52QA MS 911
Neighborhood Road
Kingston, NY 12401
U.S.A.

Notices

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 product, program, or service that does not infringe any of IBM's intellectual property rights or other legally protectible rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by IBM, are the user's responsibility.

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 Commercial Relations, IBM Corporation, Purchase, NY 10577.

Subtopics

Trademarks and Acknowledgments

The following trademarks apply to this book:

AIX is a trademark of the International Business Machines Corporation

AT is a registered trademark of International Business Corporation

Ethernet is a trademark of XEROX Corporation

IBM is a registered trademark of International Business Machine Corporation.

INed is a registered trademark of INTERACTIVE Systems Corporation

INmail and INnet are trademarks of INTERACTIVE Systems Corporation

NFS is a registered trademark of Sun Microsystems, Inc

Portions of the code and documentation were developed at th Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California under the auspices of the Regents of the University of California.

POSIX is a trademark of the Institute of Electrical and Electronic Engineers (IEEE).

PS/2 and Personal System/2 are registered trademarks of th International Business Machines Corporation.

RT is a registered trademark of the International Business Machine Corporation.

System/370 is a trademark of the International Business Machine Corporation.

Token-Ring is a trademark of International Business Corporation

UNIX is a registered trademark of UNIX System Laboratories, Inc. i the USA and other countries.

VT-100 is a trademark of Digital Equipment Corporation

VM/XA is a trademark of International Business Corporation

About This Book

The book is a planning guide for the AIX Operating System for System/370 (hereafter referred to as AIX/370). This book describes the functions and capabilities of AIX/370 and identifies the areas of planning that must be explored before installation and be continually evaluated while the system is in use. The aspects of planning discussed in this book include:

- Installatio
- Customizatio
- Networkin
- Problem diagnosi
- Operatio
- Servic
- Application programming

Subtopics

Who Should Read This Book

This book is intended for individuals responsible for purchasing, planning, or installing AIX/370.

What You Should Know

You should have enough knowledge of the hardware, software, and communications system at your site to make basic decisions about how they are used. Because AIX/370 is a guest operating system on a VM system, familiarity with VM is helpful. This book also assumes that you or someone at your site understands:

IBM hardware and software, including

- IBM System/370
- IBM System/370 XA
- IBM VM HPO
- IBM VM/SP
- IBM VM/XA SP

UNIX System V and 4.3BSD concepts and procedure

Data Communications concepts and procedures, including

- Ethernet
- Token-Ring LAN
- IEEE 802.3 LAN
- TCP/IP (Transparent Control Protocol/Internet Protocol).

How to Use This Book

You can use this book for both a general orientation to and specific instruction in planning the installation and maintenance of AIX/370. You should:

Read Chapters 1-4 of this book to become familiar with the fundamenta concepts of AIX/370 and data communications affecting installation and customization decisions. These chapters provide an overview of AIX and explore basic planning considerations for networking, TCF clusters, and AIX file systems and minidisks.

Follow the instructions in Chapters 5-7 to plan the details o installation. Chapter 6 lists the specific requirements for the AIX operating system. These chapters refer you to appendixes containing worksheets and checklists that you can use to plan and configure a TCF cluster.

Read Chapter 8 to decide what you will customize

Refer to Chapters 9-11 for information on compatibility, applicatio programming, and operation planning considerations.

Note the diagnosis and service procedures described in Chapters 12-13

A planner must be familiar with all of the material in the book, but most of the planning process is contained in Chapters 5-7. It is recommended, therefore, that the entire text be read from beginning to end at least once. If the planner understands everything that has been read, a second or third reading of the book may be restricted to Chapters 5-7.

The book also contains a glossary of terms and abbreviations.

Highlighting

This book observes the following highlighting conventions:

New terms introduced in the text are shown in ***boldface italic***.
AIX commands, options, parameters, names of keys, keywords, and actua file names are in **boldface** type.
Protocols are in UPPERCASE type
Environment variables are in **UPPERCASE** boldface type.
Variable information is in *italic* type.
Anything users type is in **monospace** type.
Anything appearing on a display screen that is referred to in paragraph of text is in **monospace** type.
Instructions set off from a paragraph are printed in **monospace** type.

Related Publications

For additional information, you may want to refer to the following publications:

AIX Access for DOS Users Administrator's Guide, SC23-2042, explains how to install and administer the AIX Access for DOS Users program on the IBM PS/2, RT, and System/370 computers running the AIX Operating System with the AIX DOS Server. It covers the responsibilities for installation, daily operation, and maintenance of the AIX Access program.

AIX Access for DOS Users User's Guide, SC23-2041, describes the AIX Access for DOS Users program and shows how to use the file services of an AIX host while running DOS applications.

AIX C Language Reference, SC23-2058, describes the C programming language and contains reference information for writing programs in C language that run on the AIX Operating System.

AIX C Language User's Guide, SC23-2057, describes how to develop, link, and execute C language programs. This book also describes the operating dependencies of C language and shows how to use C language-related software utilities and other program development tools.

AIX Commands Reference, SC23-2292 (Vol. 1) and SC23-2184 (Vol. 2), lists and describes the AIX/370 and AIX PS/2 Operating System commands.

AIX Guide to Multibyte Character Set (MBCS) Support, GC23-2333, explains the basic concepts of AIX multibyte character set support and refers to other AIX books that contain more detailed information.

AIX Library Guide, Glossary, and Master Index, SC23-2324, describes the publications in the AIX Operating System library and contains a glossary of terms used throughout the library. This book also includes a master index to the contents of each of the publications in the library.

AIX Messages Reference, SC23-2294, lists messages displayed by the AIX Operating System and explains how to respond to them.

AIX Programming Tools and Interfaces, SC23-2304, describes the programming environment of the AIX Operating System and includes information about operating system tools that are used to develop, compile, and debug programs.

AIX TCP/IP User's Guide, SC23-2309, describes the features of TCP/IP and shows how to install and customize the program. It includes reference information on TCP/IP commands that are used to transfer files, manage the network, and log into remote systems.

AIX Technical Reference, SC23-2300 (Vol. 1) and SC23-2301 (Vol. 2), describes the system calls and subroutines a programmer uses to write application programs. This book also provides information about the AIX Operating System file system, special files, miscellaneous files, and the writing of device drivers.

AIX VS FORTRAN Reference, SC23-2050, describes the FORTRAN programming language as implemented on AIX RT, AIX PS/2, and AIX/370. This book describes all of the standard features of VS FORTRAN as well as the enhanced functions and capabilities incorporated into IBM AIX VS FORTRAN.

AIX VS FORTRAN User's Guide, SC23-2049, shows how to develop and execute FORTRAN programs on AIX RT, AIX PS/2, and AIX/370. This book also explains how to compile and execute programs that contain sections of code written in the VS Pascal and C programming languages.

AIX VS Pascal Reference, SC23-2054, describes the VS Pascal programming language as implemented on the IBM PS/2 or RT with the AIX Operating System installed. This book describes all of the standard features of Pascal as well as the enhanced functions and capabilities incorporated into IBM AIX VS Pascal.

AIX VS Pascal User's Guide, SC23-2053, shows how to develop and execute Pascal programs on the IBM PS/2 and RT using the AIX Operating System. This book also explains how to compile and execute programs that contain sections of code written in the VS FORTRAN and C programming languages.

AIX X-Windows Programmer's Reference, SC23-2118, describes the X-Windows licensed program and provides information on X-Windows library functions, FORTRAN subroutines, protocols, and extensions.

AIX X-Windows User's Guide, SC23-2017, describes the X-Windows licensed program and shows how to start, run, install, and customize this program.

AIX PS/2 DOS Merge User's and Administrator's Guide, SC23-2045, shows how to use DOS in the AIX environment, including running DOS and AIX programs simultaneously and running AIX commands from the DOS environment. It also shows how to install the DOS Merge software and how to perform essential system maintenance activities, such as adding user accounts, backing up the file system, and setting up terminals.

AIX PS/2 General Information, GC23-2055, describes the AIX PS/2 Operating System's functions and capabilities and the product's position in the AIX family of products.

AIX PS/2 INed, SC23-2001, shows how to use the INed editor to create, access, and store files. This book also includes reference information on INed commands and a listing of INed error messages.

AIX PS/2 INmail/INnet/INftp User's Guide, SC23-2076, describes the INmail/INnet/INftp/Connect programs and shows how to use these programs to send mail to and receive mail from local and remote computer systems. This book also shows how to transfer files to and from other computer systems installed on the network.

AIX PS/2 Interface Library Reference, SC23-2051, contains information about the library of system calls available with IBM AIX VS Pascal and IBM AIX VS FORTRAN as implemented for use with the IBM AIX PS/2 Operating System.

AIX PS/2 Keyboard Description and Character Reference, SC23-2037, describes the characters and keyboards supported by the AIX PS/2 Operating System. This book also provides information on keyboard position codes, keyboard states, control code points, code-sequence processing, and non-spacing character sequences.

AIX PS/2 Text Formatting Guide, SC23-2044, describes the text formatting utilities available on the PS/2 and shows how to format text with NROFF and TROFF. This book also shows how to use the **vi** editor to create, revise, and store files.

AIX PS/2 Usability Services Reference, SC23-2039, lists and describes Usability Services commands.

AIX PS/2 Usability Services User's Guide, SC23-2038, shows how to create and print text files, work with directories, start application programs, and do other basic tasks with Usability Services.

AIX/370 Administration Guide, SC23-2088, describes such administrative tasks as updating the file system, backing up files, and fine-tuning and monitoring the performance of the operating system.

AIX/370 Diagnosis Guide, SC23-2090, describes procedures and tools that can be used to define and categorize symptoms of problems that may occur during daily operation.

AIX/370 General Information, GC23-2062, describes the functions and capabilities of AIX/370 and its position in the AIX family of products.

Installing and Customizing the AIX PS/2 Operating System, SC23-2290, provides step-by-step instructions for installing the AIX PS/2 Operating System and related programs. This book also shows how to customize the operating system to suit the user's specific needs and work environment.

Installing and Customizing the AIX/370 Operating System, SC23-2066, provides step-by-step instructions for installing the AIX/370 Operating System and related programs. This book also shows how to customize the operating system to suit the user's specific needs and work environment.

Managing the AIX Operating System, SC23-2293, describes such system-management tasks as adding and deleting user IDs, creating and mounting file systems, backing up the system, repairing file system damage, and setting up an electronic mail system and other networking facilities.

Using the AIX Operating System, SC23-2291, shows the beginning user how to use AIX Operating System commands to do such basic tasks as log in and out of the system, display and print files, and set and change passwords. It includes information for intermediate to advanced users about how to use communication and networking facilities and write shell procedures.

Other Publications

Here is a list of other IBM publications that may help you understand and plan your system:

VM/XA System Product General Information Manual, GC23-0362, introduces and describes the features of the IBM VM/XA SP program product. It provides information for installation managers and system programmers.

VM/SP General Information Manual, GC20-1838, introduces and describes the features of the IBM VM/SP program product. It provides information for installation managers and system programmers.

VM/XA System Product Planning and Administration, GC23-0378, is for system programmers and administrators who plan for the installation of a VM/XA SP system. It contains information on preparing statements for the system directory and on planning storage allocation.

VM/SP Planning Guide and Reference, SC19-6201, is for system programmers and administrators who plan for the installation of a VM/SP system. It contains information on preparing statements for the system directory and on planning storage allocation.

VM/XA System Product Interpreter User's Guide, SC23-0375, discusses and provides examples of how to use Restructured Extended Executor (REXX), the command language for writing procedures to run under VM/XA SP System Product Interpreter, the command processor for CMS.

VM/SP System Product Interpreter User's Guide, SC24-5238, discusses and provides examples of how to use Restructured Extended Executor (REXX), the command language for writing procedures to run under VM/SP System Product Interpreter, the command processor for CMS.

VM/XA System Product Interpreter Reference, SC23-0374, provides statement syntax, format, and usage notes for using Restructured Extended Executor (REXX), the command language for writing procedures to run under VM/XA SP System Product Interpreter, the command processor for CMS.

VM/SP System Product Interpreter Reference, SC24-5239, provides statement syntax, format, and usage notes for using Restructured Extended Executor (REXX), the command language for writing procedures to run under VM/SP System Product Interpreter, the command processor for CMS.

VM/XA System Product CP Command Reference, SC23-0358, is a reference manual for users who run systems such as OS, OS/VS, DOS, DOS/VS, VSE, CMS, and networking systems in virtual machines under VM/XA SP. It describes all CP commands.

VM/SP System Product CP Command Reference, SC19-6211, is a reference manual for class G users who run systems such as OS, OS/VS, DOS, DOS/VS, VSE, CMS, and networking systems in virtual machines under VM/SP. It describes all class G CP commands, which are commands available to general users.

IBM System/370 Extended Architecture Principles of Operation, SA22-7085, provides a detailed definition of the machine functions performed by System/370 Extended Architecture.

IBM System/370 Principles of Operation, GA22-7000, provides a detailed definition of the machine functions performed by System/370.

Table of Contents

Figures

Tables

1.0 Chapter 1. Introduction to Planning

Subtopics

1.1 Contents

1.2 About This Chapter

This chapter outlines the functions of AIX/370. AIX/370 offers most of the functions included in UNIX System V and 4.3BSD, as well as IBM enhancements, which are summarized in the following material.

1.3 AIX/370 Overview

AIX/370 is part of the AIX family of products, which includes AIX PS/2 and AIX/RT. AIX/370 is an IBM operating system that is compatible with AT&T UNIX System V and the 4.3 Berkeley Software Distribution (4.3BSD). The AIX/370 system is a transparent, distributed UNIX operating system allowing multiple computer users to be linked together. AIX/370 implements a distributed, transparent computing environment for installations that run Virtual Machine (VM). The AIX/370 system is considered a transparent, distributed operating system in that the machine boundaries are essentially invisible to users and application programs; therefore, users can access files even though the files are not stored on their own computers.

A ***TCF cluster*** is a group of computers operating under the AIX Operating System with the Transparent Computing Facility (TCF). One or more AIX/370 systems or AIX PS/2 systems can function as a cluster. Most members of the cluster communicate via a local area network (LAN). System/370 sites within the cluster may communicate with each other by other means, such as the channel-to-channel adapter (CTCA). The user interacts with the machines in a cluster as if they were a single computer, even though the actual files, disk space, and memory resources being used may be distributed all across the cluster. The AIX/370 and AIX PS/2 products with the TCF function are transparent, distributed operating systems which are compatible with each other.
AIX/370 provides:

Compliance with the IEEE standard 1003.1 for Portable Operating Syste for Computer Environments (POSIX).

The following 4.3BSD compatible functions

- Supplies a full screen editor with windowing (INed).
- The C shell which is a deluxe interactive shell with history lists, aliasing, and job control. The history function saves past commands and allows the user to re-invoke them easily. The aliasing function allows the user to make up simple names for frequently-used command sequences and then supplies the ability to re-invoke the command sequences by those names. Job control allows processes to be interactively switched between foreground, background, and also suspended.
- **dbx** - a symbolic debugging system for C source code.
- sendmail/MH - sophisticated mail programs which separate the addresses of individual messages so that complex pathnames over interconnecting networks are handled for the user automatically.
- long file name - the user is now permitted to use file names up to 255 characters long. The UNIX System V limit was 14 characters. AIX expands this to the larger 4.3BSD limit.
- Transmission Control Protocol/Internet Protocol (TCP/IP) - a communications protocol designed to allow interconnection of a wide range of different computer equipment. TCP/IP provides TELNET virtual terminal service, allowing users to log into hosts which may be of different types than their own machine, and File Transfer Protocol (FTP) for file transfers.

The following functions are common to both UNIX System V Release 2. and 4.3BSD:

- Source Code Control System (SCCS) - A system for controlling and tracking document changes (normal text and source program code).
- Provides tools for debugging and maintaining programs.
- The Bourne shell, a flexible command language that can be customized to specific applications or user needs.
- A hierarchical file system.

- Execution of processes that can be interactive, sequential asynchronous, or background (batch).
- Editors:
  - -- **ed** - original UNIX System V editor
  - -- **ex** - extended editor with macro facilities
  - -- **vi** - visual version of **ex** for full screen editing.
- Electronic mail.
- A sophisticated text processing and document preparation system.
- Networking to other UNIX systems using the **uucp** command.
- Programming tools, such as:
  - -- The C compiler - a high level programming language conducive to structured programming
  - -- **lint** - a program checker for C.

Enhancements added by IBM which include, but are not limited to, th following:

- Supports System/370 processors that are running VM/SP and VM/XA SP, including the IBM 3090 processors. This includes support for vector facility and Extended Architecture. However, no single AIX/370 Operating System can take advantage of Multi Processor (MP) support.
- Provides virtual storage support (versus swapping) on AIX/370. For System/370 AIX/370 systems, this allows for 8Mb (megabytes) of virtual space per user process independent of the real memory size or the virtual machine size. For System/370XA AIX/370 systems, this allows for 768Mb of virtual space per user process independent of the real memory size or the virtual machine size.
- Supports a heterogeneous cluster with IBM System/370 computers running AIX/370 (S/370), AIX/370 (S/370XA) and IBM PS/2s running AIX PS/2.
- Supplies VM file transfer support, using the **vucp** and **uvcp** commands.
- Provides facilities for the creation and support of replicated file systems and files. This provides the cluster environment with highly reliable replicated file systems, where changes are committed (written) to the disk at regular intervals. This increases the availability of the files and directories in the replicated file systems when some cluster sites are unavailable.
- Allows a user to start a task on a remote site. Users login to only one site in the cluster, but have the option to initiate tasks on any site. This gives users the option of choosing faster or less heavily loaded machines for their central processor unit (CPU)-intensive jobs. It also allows users to choose a machine with specific hardware features (such as a 3090 with vector support).
- Supports load leveling with commands such as **on**, **loads**, and **migrate**, and with the **fast** alias.
- Offers migration of active tasks from one site to another. A process which is already active can be moved to another processor. This allows the user to keep a task active if it is known that a certain site is due to be brought down for service. It also allows switching to a faster processor when a task is taking longer than anticipated.
- Supports TCF, which ties all sites in the cluster together so that they appear to the user as a single processor with a single set of resources. This allows the user to interact with the system as if it were one huge computer with a single large disk. The system keeps track of its own resources and the user is free to run any program against any data file for which permissions have been granted. The user does not have to know or care where the resources actually are. For example, a user might run a program

stored on Site A, process a data file kept on Site B, and the job might be performed by the CPU of Site C. Meanwhile, the user might be logged on to Site D, and the whole process would appear to be taking place on the user's machine.

- Provides an enhanced print facility to aid in spool queue handling.
- Provides extended file and record locking. Locks function transparently, so a user can lock a file located on another site in the cluster.
- Integrates Disk Operating System (DOS) and AIX via AIX Access for DOS Users and DOS Merge
- Offers X Windows Client support (X-11 Version 2).
- Provides the **onhost** facility, which allows a user on the TCF cluster to execute commands and applications on the local VM system, or on any guest operating system which is either running on the local VM or a connected to it by TCP/IP. CMS commands are supported on VM and Time Share Option (TSO) commands are supported on Multiple Virtual Storage (MVS).
- Provides ability to use AIX PS/2s for editing host files. Allows users working in a cluster environment with PCs to off-load text editing to the PCs using the DOS vi editor or other DOS text editors.
- Allows users working in a cluster environment with PS/2s running AIX PS/2 to off-load highly interactive tasks to the PS/2s with TCF.
- Allows a cluster to expand its CPU capabilities by merely adding a new CPU to the existing cluster, providing incremental growth.
- Supports the Network File System (NFS) and NFS locking, a separate Licensed Program Product (LPP).
- Provides multibyte character set support.

The TCF cluster also allows:

Communication among users in a single cluster
Interaction with other UNIX systems (such as PC/IX and XENIX) b exchanging files and data over RS-232 links.
Interaction with DOS systems through AIX Access for DOS Users
LAN connectivity between systems and users through
- Token-Ring LAN
- Ethernet LAN
- IEEE 802.3 LAN.

Subtopics

1.3.1 AIX Commands

For information about commands, system calls, subroutines, file formats, miscellaneous facilities, special files, and system maintenance procedures that are supported by AIX/370 and AIX PS/2, refer to the following manuals:

AIX Operating System Commands Reference, Volume 1
AIX Operating System Commands Reference, Volume 2
AIX Operating System Technical Reference, Volume 1
AIX Operating System Technical Reference, Volume 2.

1.3.2 AIX/370 Software

AIX/370 is comprised of two basic types of software. These are the operating system and licensed programs.

The ***operating system*** controls and coordinates how programs run in a system. It also provides services, such as resource allocation, input/output control, and data management. One of the operating systems available for the System/370 is the IBM AIX/370 Operating System. This operating system includes various optionally installable features and services.

Licensed programs are programs or a group of programs that performs a particular task. Licensed programs are optionally available products for which the user pays a license fee. The programs themselves remain the property of the licensor. The separately installable programs available to run with AIX/370 include, but are not limited to, the following:

Network File Systems (NFS
VS FORTRAN Version
Engineering and Scientific Subroutine Library (ESSL
Optimization Subroutine Library (OSL)

1.3.3 Program Compatibility

Programs written according to the AT&T Bell Laboratories UNIX System V specifications are source-code compatible, as well as most programs written for 4.3BSD. Refer also to Chapter 9, "Program Compatibility Considerations."

AIX/370 is a member of the AIX family. The AIX family provides a high degree of compatibility across several processor families. This includes the PS/2 80386-based systems, the RTs, and the System/370 class hardware. AIX/370 is highly compatible with PC/IX and IBM XENIX, and can interchange ASCII data.

Programs written for AIX/370 Version 1.2 will run on the AIX/370 Version 1.2.1 operating system as long as the data supplied to the programs does not contain multibyte characters. This compatibility exists at both the source code and binary code levels.

2.0 Chapter 2. Networking

Subtopics

2.1 Contents

2.2 About This Chapter

Each AIX/370 guest system runs under VM/SP or VM XA/SP in its own virtual machine. AIX supports an array of methods to connect AIX/370 guest system virtual machines to each other and to other computer systems outside their physical location. This chapter presents methods on how to connect AIX/370 to:

- Another AIX/370 guest syste
- Virtual machines running other VM guest
- Other VM system
- Other remote systems connected by TCP/I
- The LA
- Programmable workstation
- Files on remote system
- AIX PS/2s serving as cluster host
- DOS machines

2.3 How to Plan the AIX Network

The planner's options for connecting the AIX/370 Operating System to other machines are described in this chapter. The ways that machines can be linked together to form the distinctive network called a TCF cluster are covered in Chapter 3, "The TCF Cluster." These two chapters have a very practical purpose. At the end of Chapter 3, "The TCF Cluster," the planner will have covered the theoretical data needed to sketch out a proposed network configuration and should do so at that point. These chapters are theoretical in nature but must be read with a very practical attitude. The planner must be constantly asking, "Should I apply this facility or configuration in my own AIX/370 network?" At the end of Chapter 3, "The TCF Cluster," the planner is given directions and examples which should allow usage of the material to be applied to the cluster sketch that is being formulated.

2.4 Network Connectivity Options under AIX

The machines comprising a TCF cluster are tied together by LAN lines. The cluster hosts use these lines to communicate among themselves and with their user terminals. These LAN lines may support either Ethernet or Token-Ring protocols. In either case, the communication packets being transmitted are always assembled and passed in accord with TCP/IP protocols.

Nearly all connections between a PS/2 and computing environments outside its physical location are made via either Token-Ring or Ethernet cards. These cards are internal and are plugged into the PS/2. The PS/2 also has a multiport RS-232C adapter card which allows it to accept the asynchronous transmissions from ***fixed-function*** or ***dumb*** terminals.

The connections possible between a System/370 and computing environments outside its physical location are far more complex. Rather than internal, plug-in cards that become part of the system enclosure, the System/370 uses an assortment of separate, external hardware attachments. A number of hardware and software combinations are used.

In the following sections, the hardware and software components used to connect AIX/370 to its environment are explored. See Chapter 6, "Installation Planning" for information on specific requirements.

Subtopics

2.4.1 Connecting AIX/370 Guest System Virtual Machines to Each Other

The following components are commonly used to provide connections between AIX/370 machines. Each of them will require a unique virtual device address when configuring the AIX/370 virtual machine.

Subtopics

2.4.1.1 The IBM 3088 Channel-to-Channel Adapter Connection

This physical device transmits signals from the Input/Output (I/O) channel of one physical System/370 to the I/O channel of another physical System/370. Two AIX/370 guest system virtual machines executing on two separate System/370s may communicate by this means.

Figure 2-1 shows a channel-to-channel adapter (CTCA) joining two separate AIX/370 processors.

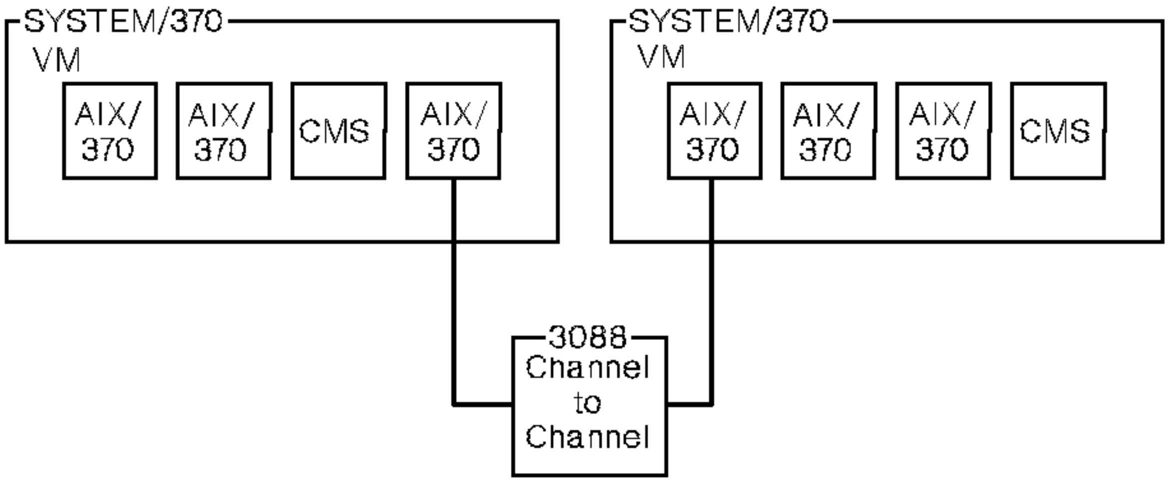


Figure 2-1. Diagram Showing a CTCA Joining Two Separate AIX/370 Processors

2.4.1.2 Virtual Channel-to-Channel Connection

The Virtual Channel-to-Channel (VCTC) connection is a software component which emulates the function of the physical CTCA device. Instead of passing communications between the physical channels of physical machines, the VCTC software passes communications back and forth between the virtual channels of the virtual machines within a single physical System/370. If multiple copies of AIX/370 are executing on multiple virtual machines within a single processor, they may communicate with each other via VCTC.

Figure 2-2 shows how VCTC connections can be made between two virtual machines running AIX/370 on the same System/370 machine.

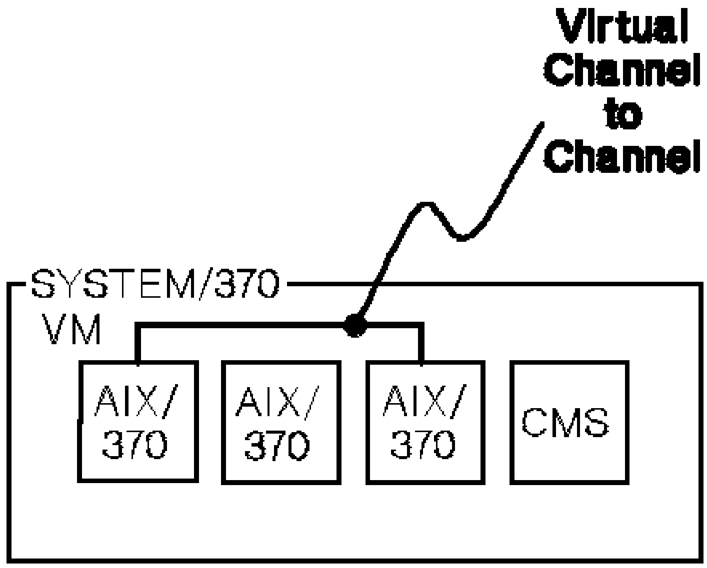


Figure 2-2. A VCTC connection communicates between two AIX/370 guest system virtual machines on the same System/370.

2.4.2 Connecting AIX/370 to Other Virtual Machines

In many cases, AIX/370 will not be the only guest running on the VM system. Several of the previously mentioned options can be used to communicate with other virtual machines on the same processor, but the **onhost** command is especially useful for communicating with other IBM guest operating systems.

Subtopics

2.4.2.1 The onhost Command

The AIX/370 **onhost** command allows users to execute processes on several other operating systems. Users can execute commands on VM/CMS or MVS/TSO, provided the desired system is accessible either on the local VM system or via TCP/IP.

The procedure involves issuing a **hostconnect** command sequence to establish connection with the desired operating system. Then, issue the following:

onhost *options command*

which executes the desired *command* on this remote system. Many of the basic AIX user commands (for example, **cat**, **who**, **rm**, and **cp**) are available.

Users can also execute CMS commands and exploit applications available under CMS (for example, PROFS, XEDIT, and FILELIST).

Figure 2-3 shows how the AIX/370 **onhost** command allows an AIX/370 user to execute commands on VM/CMS or MVS/TSO systems. For further information on the **hostconnect** and **onhost** function, refer to the *AIX Operating System TCP/IP User's Guide*.

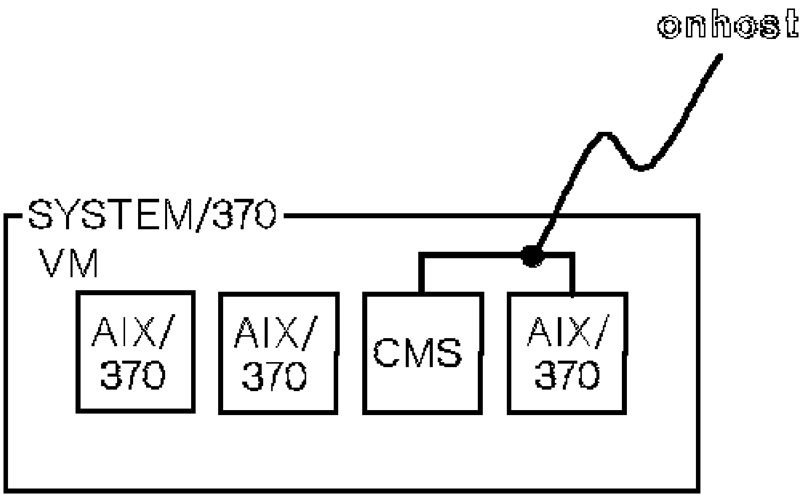


Figure 2-3. The AIX/370 onhost Command

2.4.3 Accessing VM Networking Facilities

In addition to the Real and Virtual Channel-to-Channel connection facilities just described, the AIX/370 Operating System may use the networking facilities offered by the underlying VM system, accessing those facilities by using the **onhost** command. The most convenient way of doing so is through the shell scripts created by users or by system administrators.

The **onhost** command may take advantage of such facilities as:

Pass-Through Virtual Machine (PVM
Remote Spooling Communications Subsystem (RSCS
Virtual Telecommunication Access Method (VTAM)

Subtopics

2.4.3.1 Pass-Through Virtual Machine/Inter-user Communication Vehicle

This combination allows a connection to be made between multiple virtual machines. The virtual machines may be on the same physical machine or on different physical machines.

Pass Through Virtual Machine (PVM)
This is a VM program which allows the user to open a computing session on a different machine. That is, it allows the user to ***pass through*** from one virtual machine to another. PVM runs in its own service virtual machine.

Inter-user Communication Vehicle (IUCV)
This is a VM option which allows the user to establish a communication path between two virtual machines within the same physical machine. This option must be authorized in the VM directories of both virtual machines.

Figure 2-4 shows how the PVM/IUCV combination is used to enable communication between virtual machines on different processors.

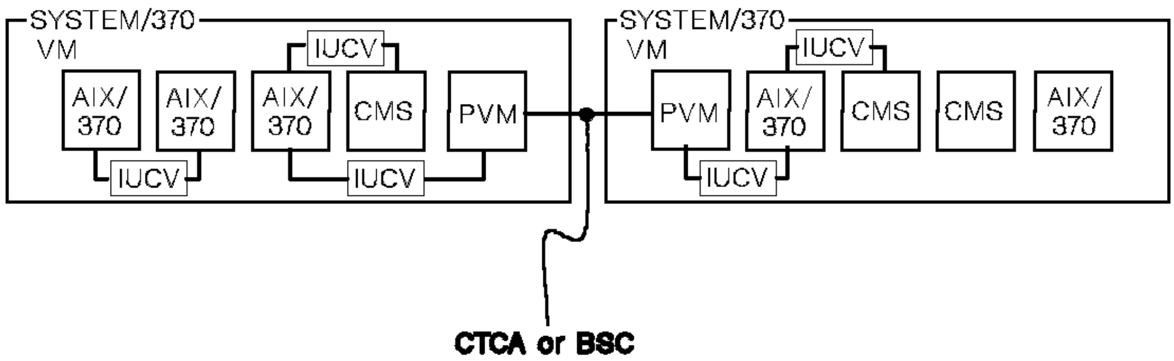


Figure 2-4. IUCV connects virtual machines on the same System/370. PVM/IUCV connects virtual machines on different processor complexes.

2.4.3.2 Remote Spooling Communications Subsystem (RSCS)

The Remote Spooling Communications Subsystem (RSCS) program runs in its own servo-virtual machine. It performs file transfers to users on other guest operating systems on the same CPU or on other machines available through VM networking channels.

Figure 2-5 shows an AIX/370 virtual machine connected to a CMS system via RSCS. These two systems are running on separate CPUs, but the same type of link can be made between two systems running on the same System/370.

This type of link is common for connecting the **mail** facility of the AIX/370 Operating System to the CMS NOTES facility of CMS.

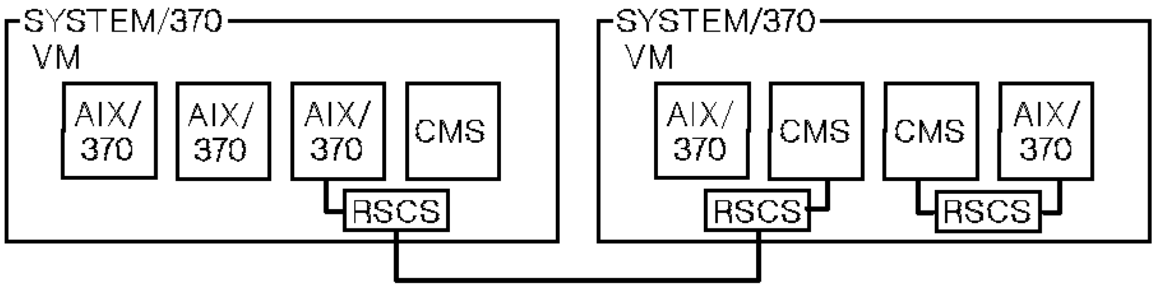


Figure 2-5. Diagram Showing an RSCS Link between AIX/370 and CMS

2.4.3.3 Virtual Telecommunication Access Method (VTAM)

Virtual Telecommunication Access Method (VTAM) is a program set developed by IBM to manage telecommunications processing. VTAM controls communications between terminals and application programs running under VSE, OS/VS1, OS/VS2, VM, and MVS. VTAM allows access to a variety of applications installed on the same CPU.

2.4.4 Connecting AIX/370 Across Distances

Figure 2-6 illustrates one possible method of connecting AIX/370 guest system virtual machines which are executing on physical processors separated by a considerable distance.

When large numbers of users are involved, a high-throughput link is recommended; in Figure 2-6, it is a T1 line. Processors separated by even greater distances may employ a satellite link in place of the T1 line.

System Network Architecture (SNA) is a multi-layered data communications system which is very similar to the Open Systems Interconnection (OSI) model. There are seven layers, one of which is the Data Link Control Layer, for which the actual protocol is Synchronous Data Link Control (SDLC).

The following may be used to implement a SNA network connection:

VM TCP/IP
TCP/IP has been ported to many systems. In Figure 2-6, VM TCP/IP is being used to pass data from AIX/370 to VTAM.

Synchronous Data Link Control (SDLC)
SDLC is a bisynchronous data transmission protocol responsible for the transmission of data between two nodes across a physical link. In Figure 2-6, the medium being used is a T1 line.

Network Control Program (NCP)
NCP is a software product usually run on the 37x5 device. Its function is to connect VTAM to a network link.

T1 lines
These are high throughput telephone lines capable of multi-band carrier function. They require modems at each end to transform their signals from the format put out by the processor to the format used by the T1 line. In Figure 2-6, the modem function is being performed by the IBM 37x5.

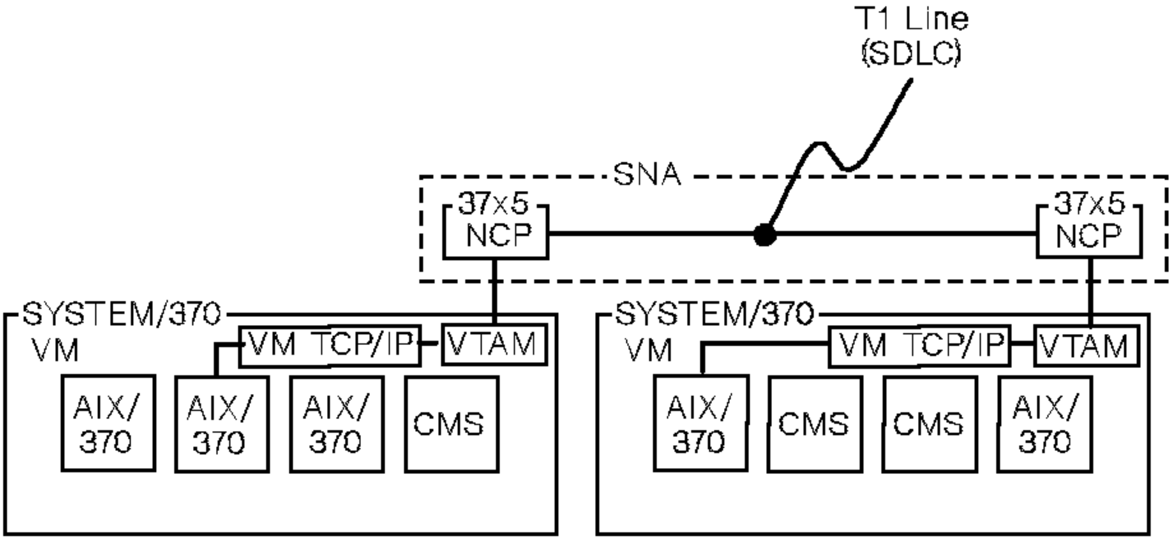


Figure 2-6. Diagram Showing a SNA Network Connection between AIX/370 Systems that Are Separated by Large Distances

2.4.5 Connecting the AIX/370 Guest System Virtual Machine to the LAN

In the TCF Cluster, the LAN is the backbone for most communication.

Two AIX/370 guest system virtual machines that are included in the same cluster may communicate via the LAN. The LAN is the channel of communication between an AIX/370 guest system virtual machine and the AIX PS/2s which may be included in most TCF clusters. The LAN is also the way the AIX/370 Operating System communicates with its user display stations.

Connecting an AIX/370 virtual machine to the LAN is accomplished by:

8232 LAN Channel Adapte

9370 Integrated LAN Adapter

Figure 2-7 shows these devices connecting an AIX/370 guest system virtual machine to the LAN.

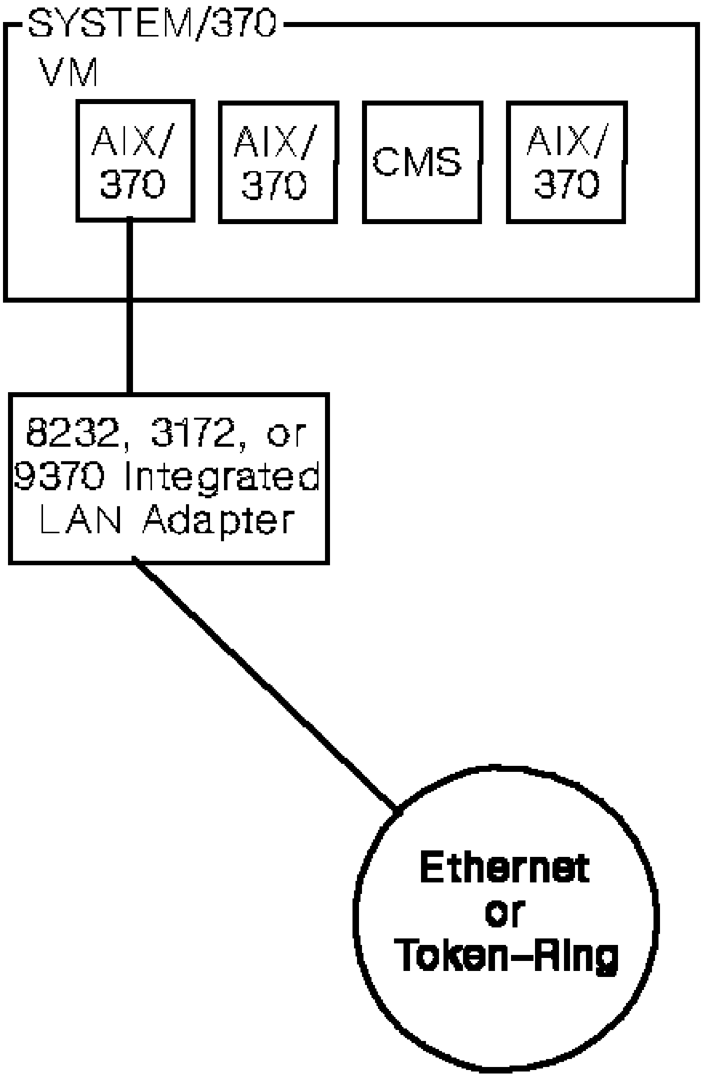


Figure 2-7. AIX/370 Guest System Virtual Machine Connected to the LAN

Subtopics

2.4.5.1 The IBM 8232 LAN Channel Station

This hardware device makes the necessary connection between the I/O channel of a System/370 and the LAN.

An 8232 is an IBM PC AT being used as a LAN adapter. Ethernet or Token-Ring cards are plugged into the device on one side, and a channel connector on the other side. The 8232 is plugged into the I/O channel of a System/370; it then allows the LAN to be read from, or written to, as if it were a channel-attached device.

The LAN traffic may be separate communications coming in directly from those users who have TCP/IP software running on their workstations. But often, it will be TCP/IP packets coming from users by way of the PS/2s of the cluster, which are providing terminal ***concentrator*** services. The user terminals may have made contact with the PS/2s by some means entirely different from TCP/IP.

The 8232 channel station is supported by VM TCP/IP, MVS TCP/IP, and AIX/370.

2.4.5.2 The IBM 9370 Integrated LAN Channel Adapter

The 9370 processors use the Integrated LAN Adapter to accomplish the same function as the 8232. The 9370 Integrated LAN Adapter requires one of the following subsystem controllers:

IEEE 802.3 LAN Subsystem Controller for Ethernet (#6035

IBM Token-Ring Subsystem Controller (#6135)

2.4.6 Connecting Workstations to AIX/370

The AIX/370 Operating System does not provide facilities for receiving the asynchronous serial transmissions produced by fixed-function terminals. Therefore, any display station used to access AIX/370 directly must do so across the LAN. This means that it must be capable of executing TCP/IP protocols.

The software packages shown in Figure 2-8 are used to provide connections between AIX/370 and the user environment. The two qualities they all share is that all run across the LAN and all are based on TCP/IP. First, there is TCP/IP itself, which is the fundamental protocol used by the LAN. Then, there are a series of packages built upon that base.
Figure 2-8 shows three software packages that are used to connect ***programmable*** or ***intelligent*** workstations to AIX/370.

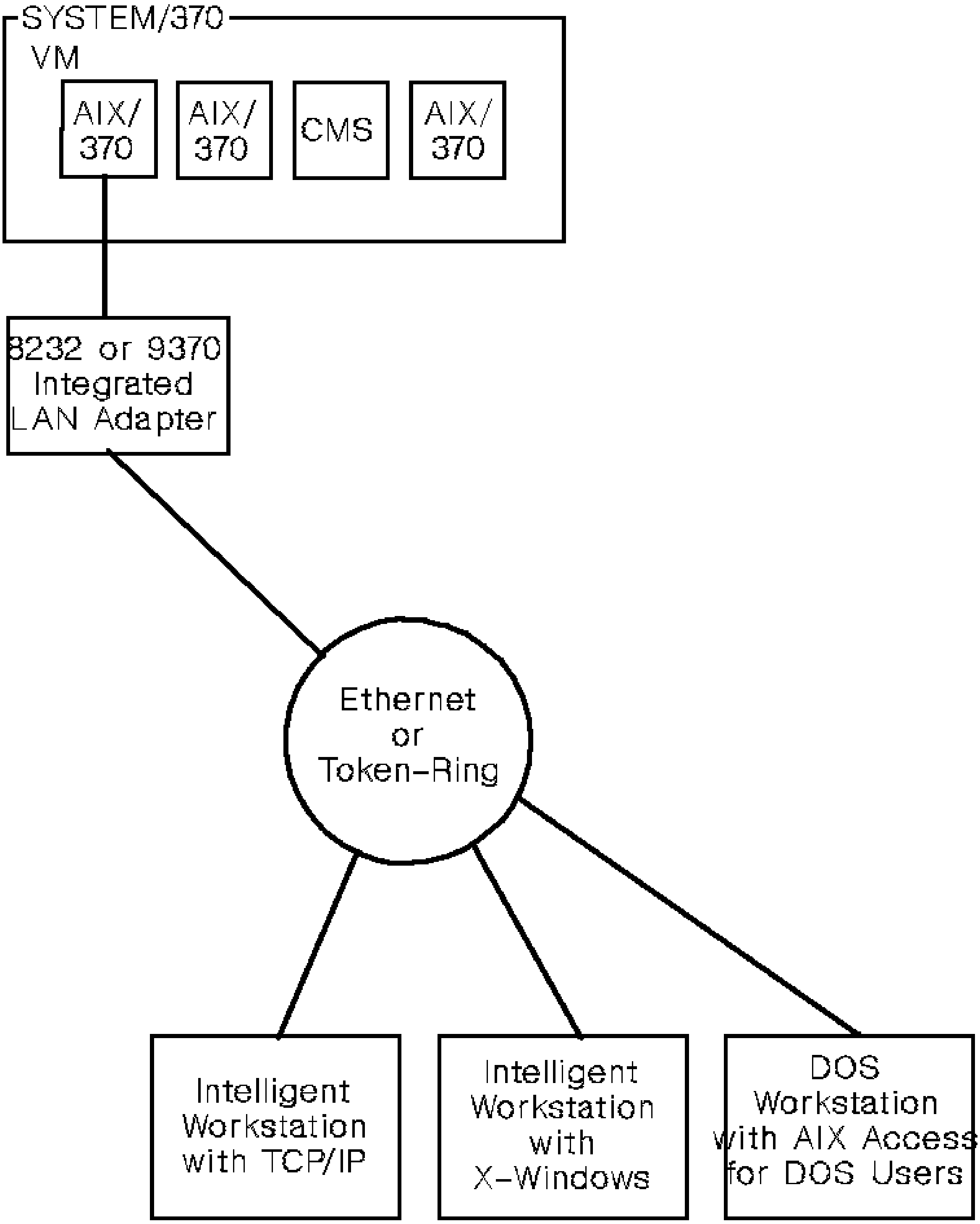


Figure 2-8. Software packages can be used to connect programmable workstations to AIX/370.

Subtopics
2.4.6.1 Transmission Control Protocol/Internet Protocol (TCP/IP)
2.4.6.2 X-Windows
2.4.6.3 AIX Access for DOS Users

2.4.6.1 Transmission Control Protocol/Internet Protocol (TCP/IP)

Transmission Control Protocol/Internet Protocol (TCP/IP) is an industry-standard communications protocol designed to connect a wide variety of computers. It is used by non-IBM machines and supported on a broad spectrum of hardware platforms. Executing TCP/IP gives AIX/370 access to a vast range of possible interconnections, especially in the technical and scientific community.

TCP/IP provides the user access to the following functions:

telnet Users may log in to one host of the network and start a remote shell on another host, running a full UNIX session on that second host.

FTP Users may transfer files back and forth between hosts in the cluster and between the cluster and those remote machines which are also executing TCP/IP.

rexec Users may execute single commands on a remote system.

rsh Users may initiate a remote interactive shell on the remote system.

The planner must account for the selection and assignment of unique TCP/IP addresses for every LAN interface in the configuration. TCP/IP is supported by both AIX, VM, MVS and DOS. VM support for AIX/370 requires the VM TCP/IP Program Product (5798-FAL).

2.4.6.2 X-Windows

X-Windows is a software product which allows the user to conduct multiple computing sessions with each session appearing on the workstation display in its own separate window.

X-Windows functions on a server/client relationship. In X-Windows terms, the ***server*** is the software program that drives the display; the ***client*** is an application program which sends a stream of data to the server.

Both AIX and DOS support an X-Windows server. In AIX, it is called X-Windows; in DOS it is called PC X-Sight.

This server software executes on the user's own machine and drives the output to the display. The corresponding client software must be running on the machine where the user's process is running. Since the user may be running multiple sessions on multiple machines, the user's server may be handling the signals from many different clients on different machines, and assembling a single screen image from this flow of data.

2.4.6.3 AIX Access for DOS Users

AIX Access for DOS Users is also referred to as PC Interface. This licensed program product is available for PCs and PS/2s (running DOS 3.3). It permits these computers to access data on the AIX Operating System and to use the AIX printers.

AIX Access for DOS Users also allows a DOS machine (or a PS/2 running DOS) to connect with a System/370 machine by either an asynchronous line (RS-232) or LAN line (running TCP/IP). The software allows a DOS PC to emulate the widely used DEC VT-100 ASCII terminal and to conduct a standard UNIX session.

The DOS machine can also read and write to the AIX file system as if those files were present on the PC's own disk. Files created this way in the AIX file system can be accessed by other AIX users or by programs running under AIX.

AIX Access for DOS Users also provides a DOS-based version of the standard UNIX editor, vi, which allows editing to be offloaded to DOS machines. The effect is to convert available DOS machines into fully functional elements of the AIX network.

The hosts of the TCF cluster must be executing a corresponding software element called DOS server. This is a built-in component of AIX/370 and an optional licensed program product for AIX PS/2.

The AIX PS/2 Operating System executes on the IBM PS/2 Models 55, 70, and 80. The PS/2 can have its fixed disk partitioned to hold DOS files in one section and AIX files in another. In this way, it can be booted up as either an AIX or DOS machine.

Note: AIX Access for DOS Users does not provide multibyte character set support.

2.5 Connecting AIX/370 to Files on Remote Systems

The purpose of networking is to create a wider sphere of user-access to processing resources, special features, and, most of all, to data. AIX offers many ways to access files stored on remote systems. Several have already been covered. This section deals with two systems; one for transfer of files to and from remote systems, and one which allows the user to treat remote files as if they were local.

Subtopics

2.5.1 Basic Network Utilities (BNU/uucp)

Basic Network Utilities (BNU) is a set of programs drawn from the UNIX tradition, where it is better known as **uucp**.

Other UNIX systems include the **uucp** command, which allows AIX to share data among these systems. The BNU protocols will run across ordinary telephone lines and provide a certain level of system security.

The electronic mail facility, which is built into AIX, automatically takes advantage of any **uucp** links that exist, thus widening the range of potential communication for users.

The AIX/370 Operating System does not execute the **uucp** command directly. The normal method is to use an AIX PS/2 machine in the same cluster to support the **uucp** connection. As you will see in Chapter 3, "The TCF Cluster," this is a standard AIX practice for many kinds of network connections.

2.5.2 Network File System (NFS)

Network File System (NFS) is a software package which allows AIX to share file systems with remote systems across network lines. By doing this, machines separated by large distances (and even machines running different operating systems) can access a common pool of files as easily as they can use their own local files. This allows them to act, in some respects, as if they were a single machine.

The primary advantage of NFS over other remote file access methods is that it is ***file sharing*** rather than ***file transfer***. The files remain in place on the remote system; they are not copied to AIX, modified, and then copied back to the remote system. A copying system leads to synchronization problems in cases where two parties are updating the same file at the same time. In NFS, all changes take place across the network lines and there are no synchronization problems.

NFS was integrated from Sun Microsystems and has been ported to AIX, VM/CMS, and MVS/TSO. However, the package is implemented differently in these various ports.
In the non-AIX operating systems, NFS functions in a server-only mode. Through NFS, VM TCP/IP can make a file system available to the outside computing environment, but no access to the files of the outside computing environment are gained by the system acting as server. These VM files can only be accessed by an outside system which is running some implementation of NFS that offers a client mode (like AIX or other UNIX systems). The VM implementation of NFS gives the VM users no ability to access files outside their own system.

In the AIX environment, NFS is a server/client system which grants ***two-way file sharing***. A system which acts as server, offering its files to the outside computing environment may also gain access to the file systems of those who act as clients.

Figure 2-9 shows two ways for AIX/370 to access files on remote systems.

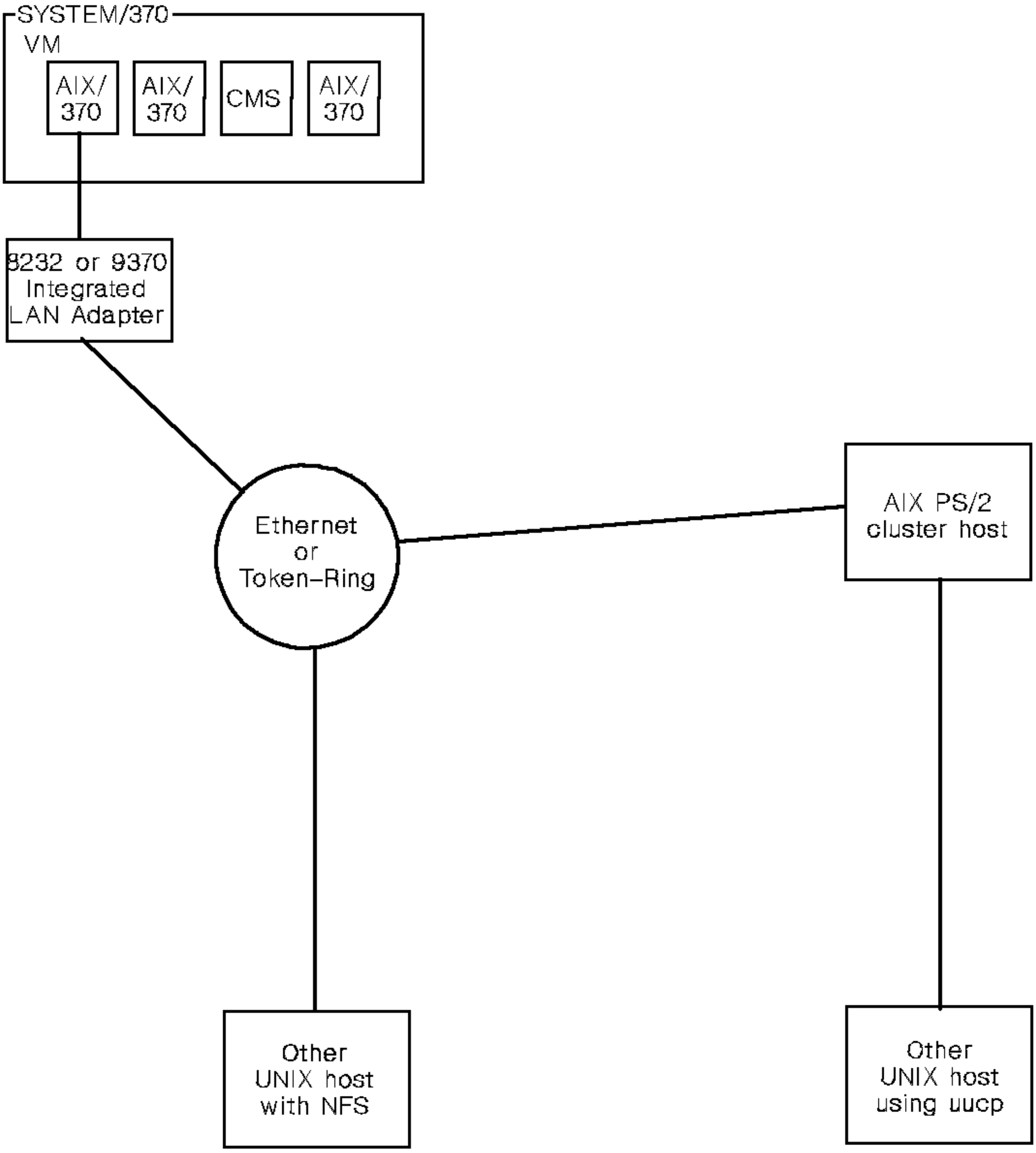


Figure 2-9. Diagram showing AIX/370 accessing files on remote systems.

2.6 Connecting AIX PS/2 hosts to AIX/370

Since AIX/370 does not have direct RS-232 connection with ASCII terminals, most clusters will include PS/2 machines running the AIX PS/2 Operating System. AIX PS/2 machines may be used as terminal concentrators in TCF clusters.

All communication between the cooperating hosts of such a cluster take place across the LAN. LAN communication is the heart of the TCF cluster, which is fully discussed in Chapter 3, "The TCF Cluster." The PS/2 may play several of these roles at once.

The PS/2 workstation has three possible roles in a TCF cluster:

It may function as one of the hosts of the cluster

It may function as a terminal concentrator if fixed-function terminal are attached to the cluster.

It may function as a programmable workstation connected to the cluste across the LAN.

Figure 2-10 shows the multi-usage applications for the AIX PS/2 machines.

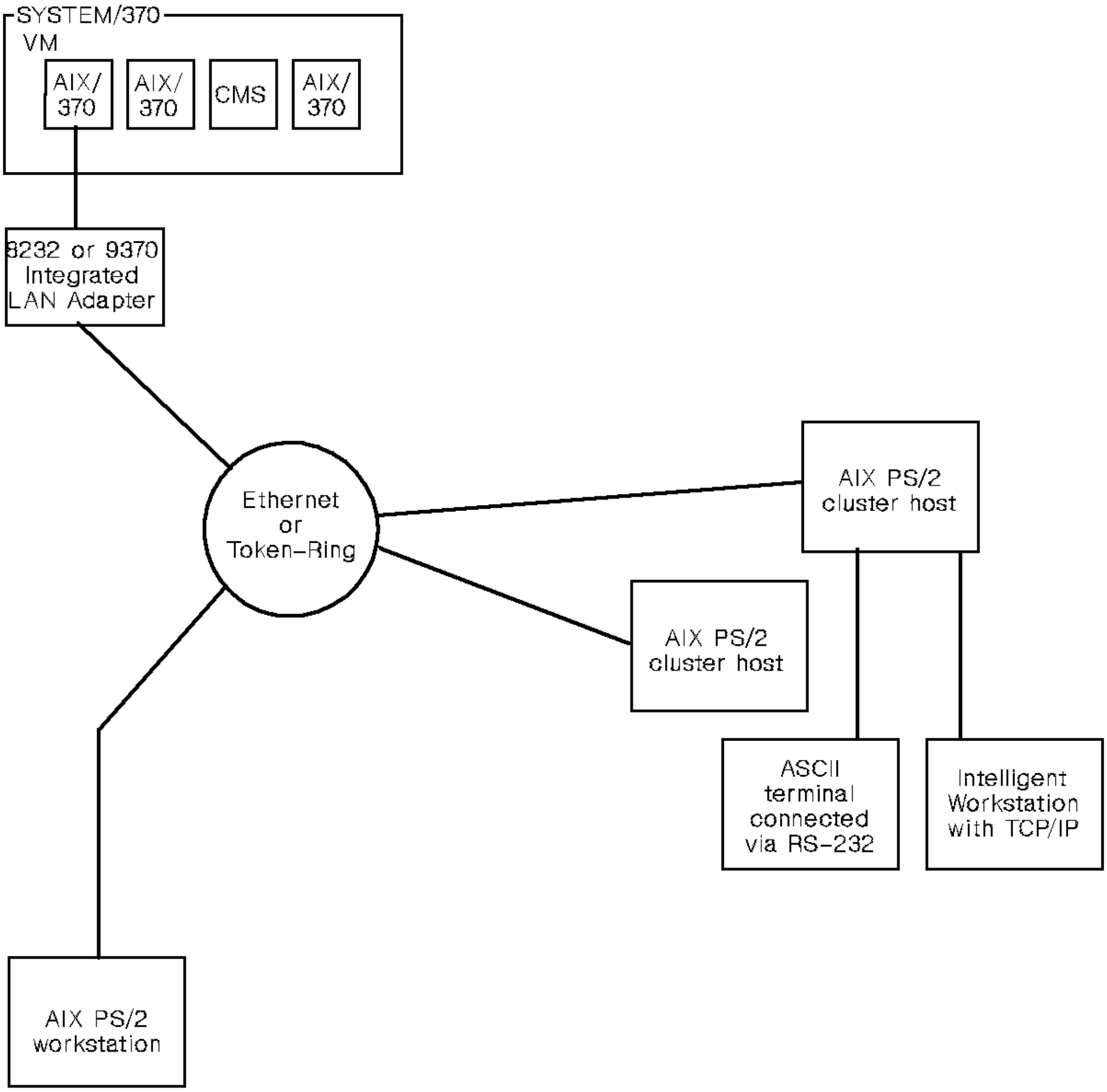


Figure 2-10. AIX PS/2 machines can serve as programmable workstations, cluster hosts, or terminal concentrators.

In its role as a cluster host the PS/2 uses only TCP/IP. All communication between the AIX/370 and the AIX PS/2 Operating System takes place across the LAN.

In its role as a terminal concentrator, the PS/2 workstation may communicate with its terminals by either of two possible means:

TCP/IP protocol

- Used to receive transmissions from programmable workstations
- Received by an Ethernet or Token-Ring card installed in the PS/2.

RS-232C protocol

- Used to receive transmissions from fixed-function terminals or PCs emulating fixed-function terminals
- Received by an asynchronous RS-232 card installed in the PS/2.

The LAN is the common bonding medium for the TCF cluster. When PS/2 workstations are used as hosts in an TCF cluster, they can execute a number of software packages that will allow them to exchange data with programmable workstations. The LAN is common to all of these software packages. They must all use TCP/IP protocols to send their packets across the Ethernet or Token-Ring lines.

The following software packages allow programmable workstations to access AIX PS/2 machines:

AIX Access for DOS Users (PC Interface

X-Windows (offering bit-mapped graphics

TCP/IP (offering **telnet** and FTP).

In addition to the above, the following two software packages offer special services:

NFS (offering remote file sharing

TCF (offering name and process transparency and migration)

Most of this is entirely transparent to the user and is simply available on the logon PS/2 hosts. The user will perceive these functions as simply another quality of the cluster.

Subtopics

2.6.1 Connecting PS/55 and 5550 Users to AIX/370

The PS/55 is the Japanese language equivalent of the standard PS/2 while the IBM 5550 is the Japanese language equivalent of the standard XT-class machine. Both of these machines offer a Japanese language keyboard and the ability to display Kanji fonts on the screen.

The PS/55 displays Kanji by running X-Windows; it is normally attached to the LAN by an Ethernet or Token-Ring connection. The 5550 has Kanji fonts built into ROM and can be attached to any 386 network host via a serial line. This host may be either a PS/2 or a PS/55 and functions as terminal concentrator for user workstations.

The PS/55 can also be used as an intelligent workstation in a scientific or technical environment allowing users to run CPU-intensive engineering applications.

If the cluster is a mixed-language network, with both Japanese and European languages in use, the machine to which these user workstations are attached can be either a PS/2 or a PS/55. The advantage of the PS/55 is that it allows Japanese language entries at its console.

If the entire cluster is exclusively Kanji-configured, the System/370 machine also requires a console unit (a PS/55 running PS/55 as 3270 software) which can function in Kanji.

The System/370 machine with which Kanji users interact needs to be provided with AIX/370 Version 1.2.1 software. If Kanji characters are to be printed on the VM system printers, a Kanji-capable version of VM/XA SP is necessary.

Figure 2-12 in topic 2.6.3 shows a typical connection of a PS/55 and 5550 to an AIX/370 System.

2.6.2 Connecting DOS Machines to AIX/370

DOS machines can be connected to the TCF cluster through any of the following software packages:

AIX Access for DOS Users which allows a DOS machine to connect

- To AIX/370 via an AIX PS/2 host in the same cluster
- Directly to AIX/370 across the LAN.

NFS allows a DOS machine to access remote files which the AIX/37 Operating System has mounted by use of NFS software. These files appear to be part of the local cluster file system.

X-Windows allows a DOS machine running PC X-sight to displa bit-mapped graphics generated by the AIX/370 Operating System.

TCP/IP provides a DOS machine a full range of **telnet**, **FTP** file transfer services, **rexec**, and **rsh**.

Figure 2-11 shows how DOS machines may serve as programmable workstations or emulate ASCII terminals.

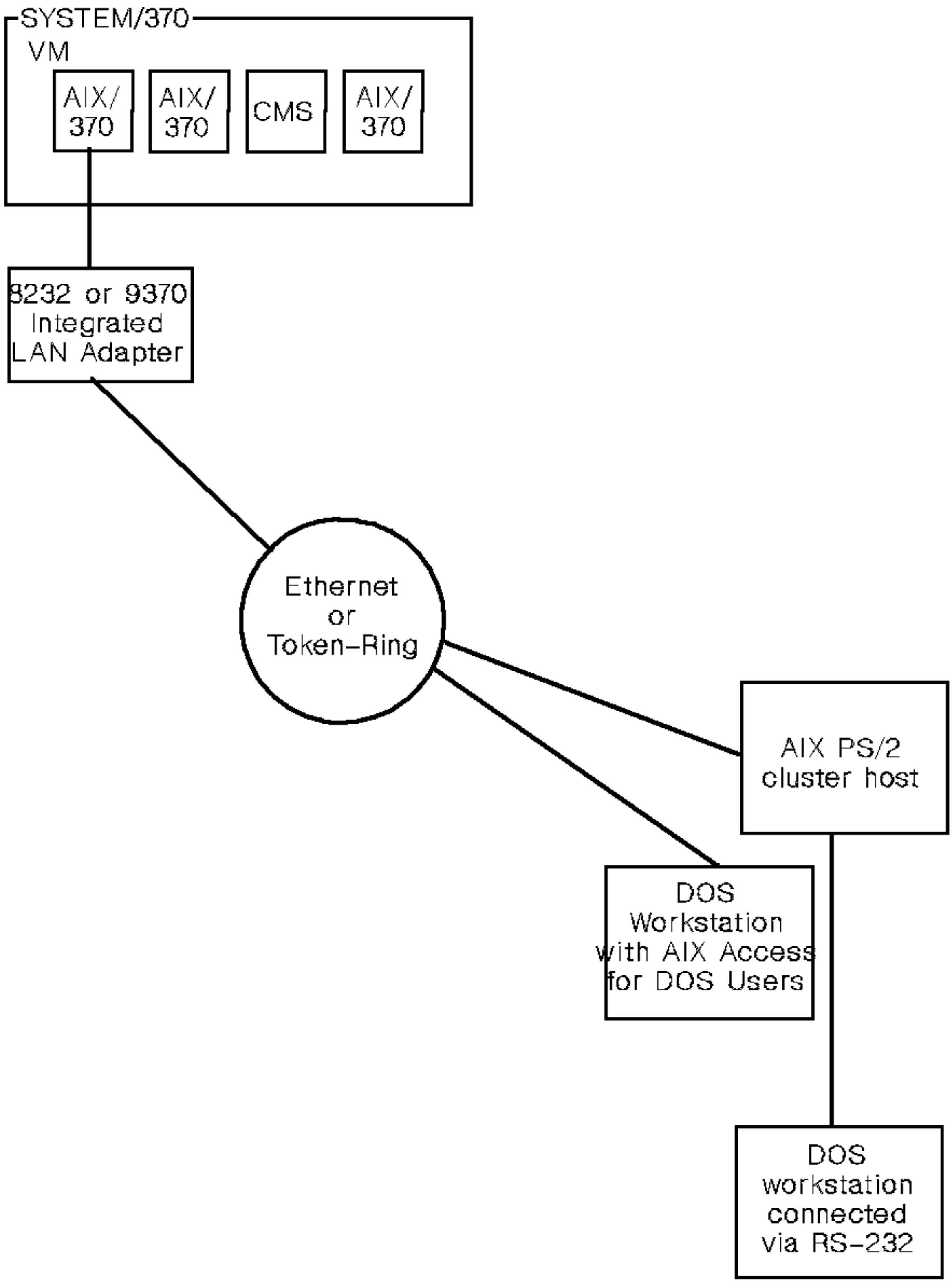


Figure 2-11. DOS machines may serve as programmable workstations or emulate ASCII terminals.

2.6.3 A Complete AIX/370 Operating System

In some installations, AIX/370 may serve hundreds of users. A number of AIX/370 Operating Systems may be in use, running on a number of different processors.

In many applications there will be AIX PS/2 machines clustered with System/370 machines, both serving as terminal concentrators. Each installation will try to take advantage of existing equipment and each will serve groups of users with different needs.

AIX/370 is a flexible system. The possible combinations of hardware and software that may be used are numerous. An installation of such complexity requires very careful planning.

Figure 2-12 shows the options that can produce AIX/370 systems of great complexity and flexibility.

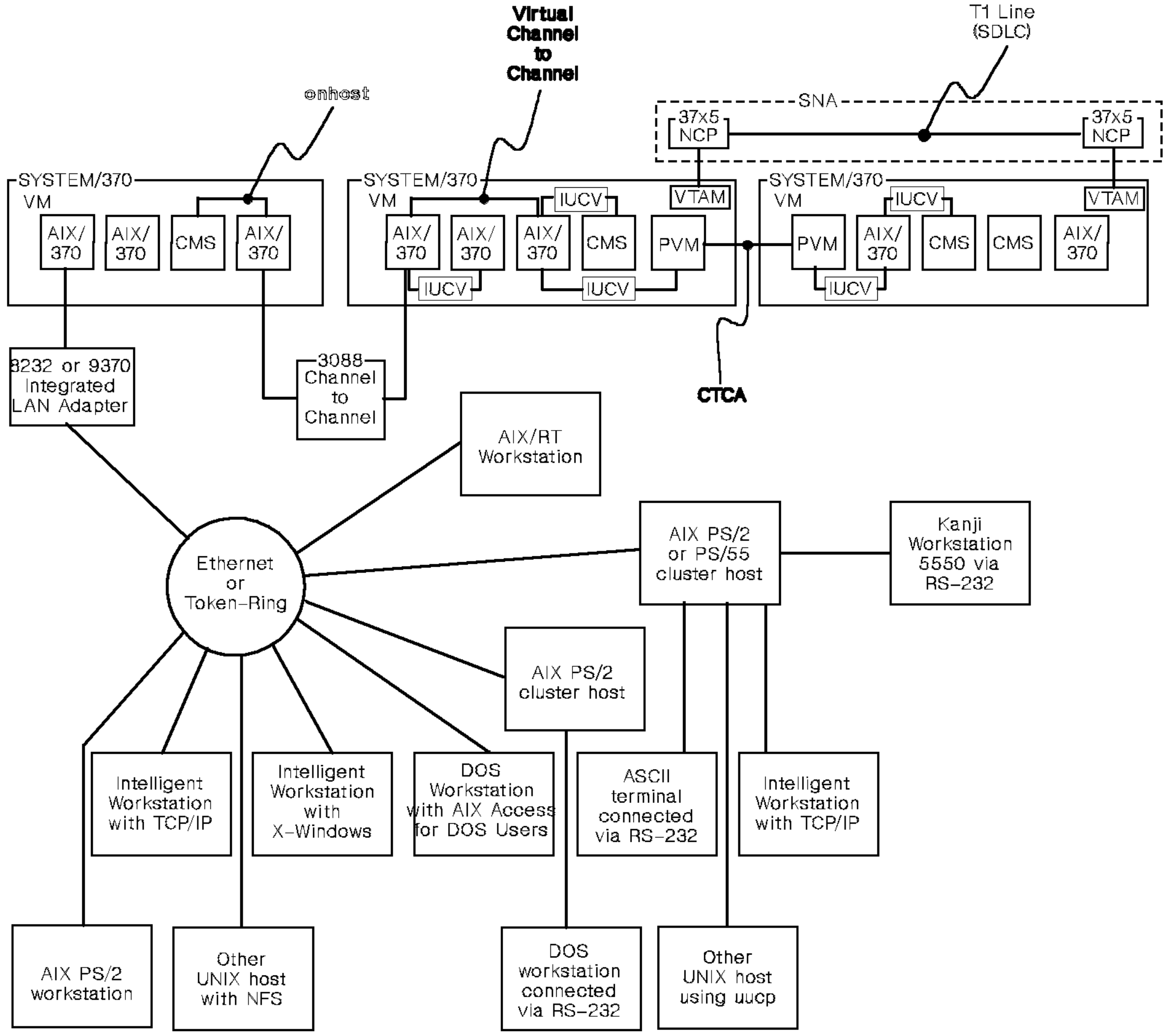


Figure 2-12. Hardware and software options can produce an AIX/370 system of great complexity and flexibility.

3.0 Chapter 3. The TCF Cluster

Subtopics

3.1 Contents

3.2 About This Chapter

This chapter describes the advantages TCF provides AIX users. Advice is presented on planning for and building the cluster. Diagrams illustrate the types of clusters that may be constructed and how to arrange user terminals. The planner is given data on how to balance processor loading, file access and editing usage in the cluster. Diagrams illustrate some actual clusters that can be built and the chapter ends with factors that may restrict cluster performance.

3.3 TCF Cluster Overview

From a user's point of view, Transparent Computing Facility (TCF) is an appealing feature of AIX functional ability. The user can interact with a large and complex network as if it were one file system. The operating system does nearly all of the network maintenance operations.

The system ensures that the proper program module is run for the processor type being used. The system keeps track of which file is kept on which disk of which network host.

The user need only know the command name, the name of the data file to process, and the meaning of the flags which represent the possible options that may be employed. Chapter 4, "AIX File Systems and Minidisks" defines the file system manipulations by which this is accomplished. This chapter covers the way the physical processors are connected together to form a cluster, and the way user terminals are attached.

Note: This chapter does not include specific requirements for TCF. See Chapter 6, "Installation Planning" for more information.

Subtopics

3.3.1 Special Definitions

Cluster
A group of AIX machines that are connected together. This cluster has the ability to function as a single machine that presents the user with a consistent interface and a consistent set of resources that looks the same from any position in the group.

Host
Any of the cooperating AIX machines in the cluster; each host can serve many users.

Note: The VM system administrator should note that this usage of host is distinctly different from the VM definition of the same word.

3.3.2 Transparent Computing Facility (TCF)

TCF is a form of file sharing which allows the hosts of the cluster to present the user with a single-system image. All the files of the cluster appear to be grouped into a single hierarchy and to be fully available to users as if the files were physically located on their own machine.

TCF provides the following special features:

Process Execution Transparency
A process which can only execute on a System/370 machine can be called for when logged onto a 386 machine. The process will be automatically executed on the System/370 with no further action required from the user.

Process Migration
A process which has been started on one host of the cluster can be stopped and restarted on another.

Name Transparency
Each available program on the system has just one name. If separate execution modules are necessary for different processors, the system keeps track of the multiple module names. Users issue a single command, no matter which type of processor they are logged onto.

Subtopics

3.3.2.1 Building the Cluster

A cluster is comprised of the various host machines and their attached user terminals.

The first consideration in building a cluster is to choose the host configuration. The planner must choose which types of machines are to be connected together, their physical location, the hardware and software links by which they are to be interconnected, as well as the roles they will play in the cluster.

The second consideration is the user terminals. The planner must choose the number and type of terminals to be used, where they are to be situated, and how they will be connected to the cluster.

This decision-making process must take into consideration the physical layout of the work area, the number of users to be served, and what existing equipment is to become part of the new AIX Operating System.

3.3.3 Types of TCF Clusters - Host Configurations

TCF clusters can be formed from different combinations of machines. All of the following examples of clusters will run the TCF option. This permits them to share files in such a way that they present the user with a single-system image.

Subtopics

3.3.3.1 The Homogeneous PS/2 Cluster

This is a group of 386-based machines clustered together. Up to 31 such machines may be clustered together. All the hosts of such a cluster will be PS/2s. Normally there will also be user terminals attached to one or more of the hosts. Each PS/2 Model 80 can support up to 16 users. All hosts of the cluster must run AIX PS/2, with the TCF option.

Figure 3-1 shows a homogeneous PS/2 cluster.

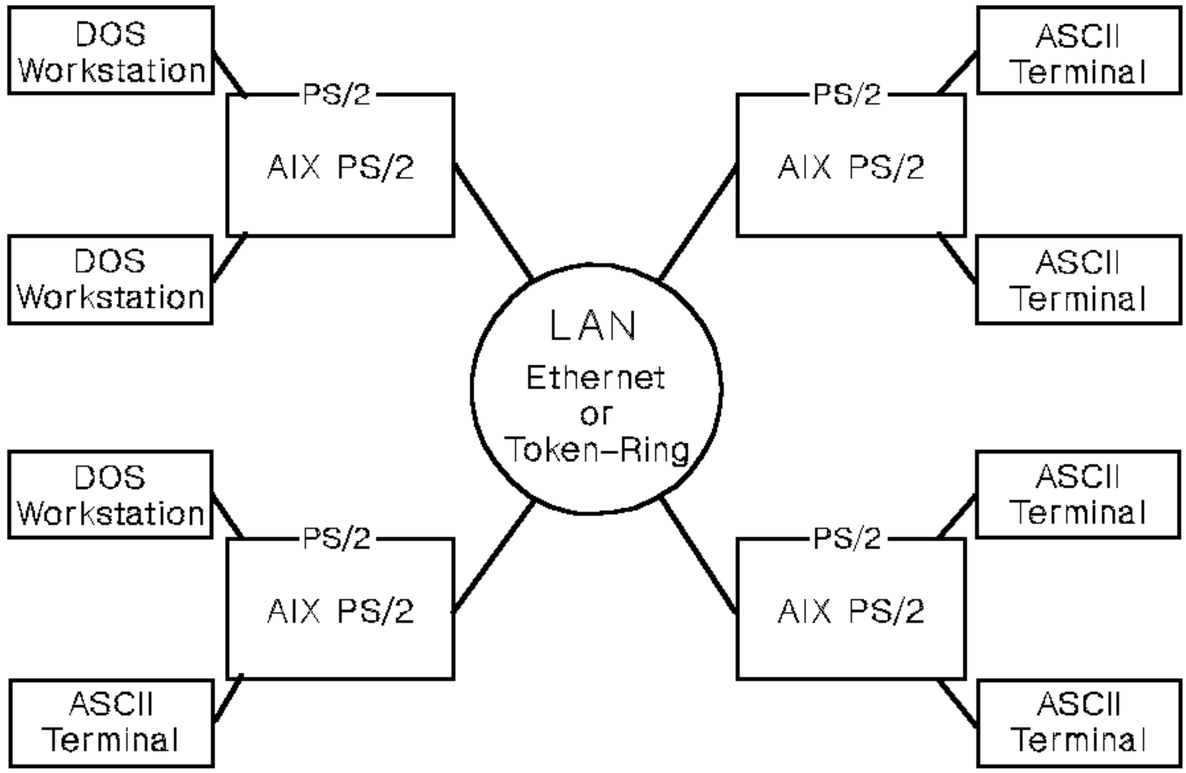


Figure 3-1. Diagram of Homogeneous PS/2 Cluster

3.3.3.2 The Homogeneous AIX/370 Cluster

Two or more System/370 machines can be linked together as a cluster. These can be AIX/370 guest systems on separate physical machines. They could also be AIX/370 guest systems executing on different virtual machines within the same physical System/370 machine.

AIX PS/2 has the capacity to accept asynchronous transmissions from ASCII terminals as well as LAN transmissions. The AIX/370 Operating System accepts only LAN transmissions.

Therefore, since there are no PS/2s in this cluster to act as terminal concentrators for the asynchronous transmissions from ASCII terminals, this configuration requires that all user terminals be programmable workstations. They must run TCP/IP and connect with the hosts of the cluster via the **telnet** or **rlogin** facilities.

Figure 3-2 shows a homogeneous AIX/370 cluster.

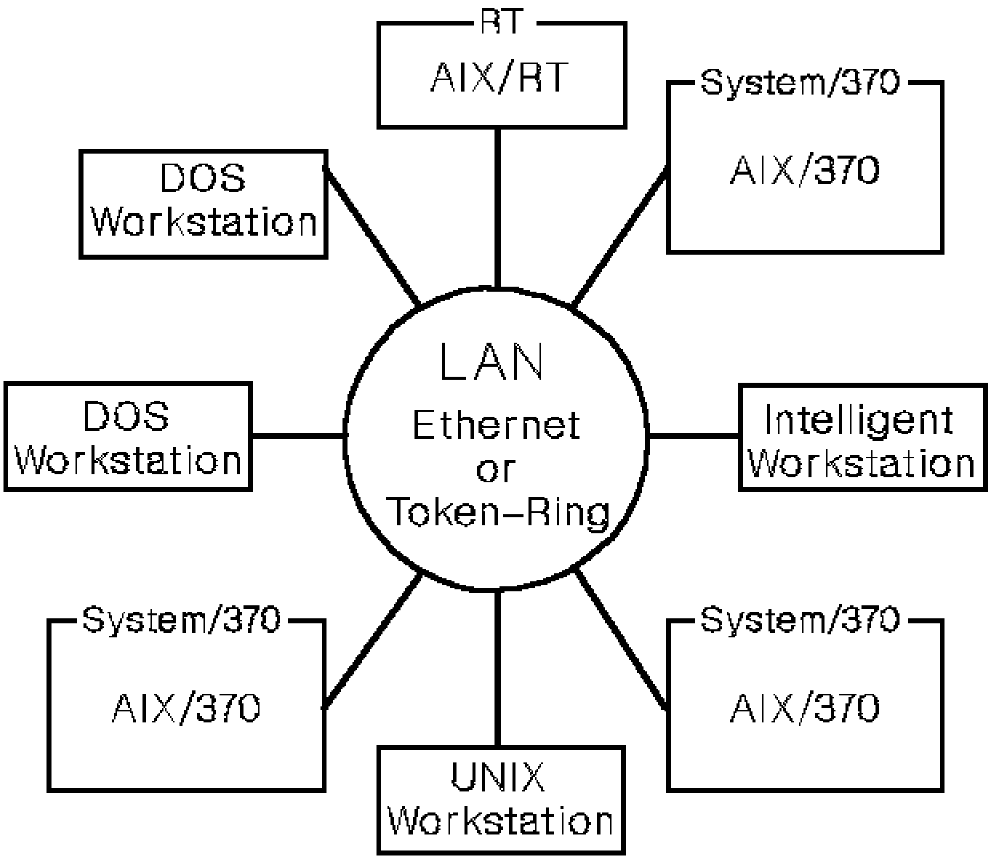


Figure 3-2. Diagram of Homogeneous AIX/370 Cluster

3.3.3.3 The Heterogeneous Cluster

This configuration clusters one or more AIX/370s with one or more PS/2s. User terminals are attached to the PS/2s, either by asynchronous serial line or by LAN line. Up to 31 hosts may be clustered together as a group. One of the functions of the PS/2s in this configuration is to act as terminal concentrators, though they also perform other data processing jobs. AIX/370 machines in such an arrangement can handle heavy, CPU-intensive processing.

One of the AIX/370s will always be chosen to function as the primary site. This machine will be responsible for storing the main copy of the shared replicated root file system. For more information on the replicated root file system, refer to Chapter 4, "AIX File Systems and Minidisks" and the *AIX/370 Administration Guide*.

Figure 3-3 shows a heterogeneous TCF cluster.

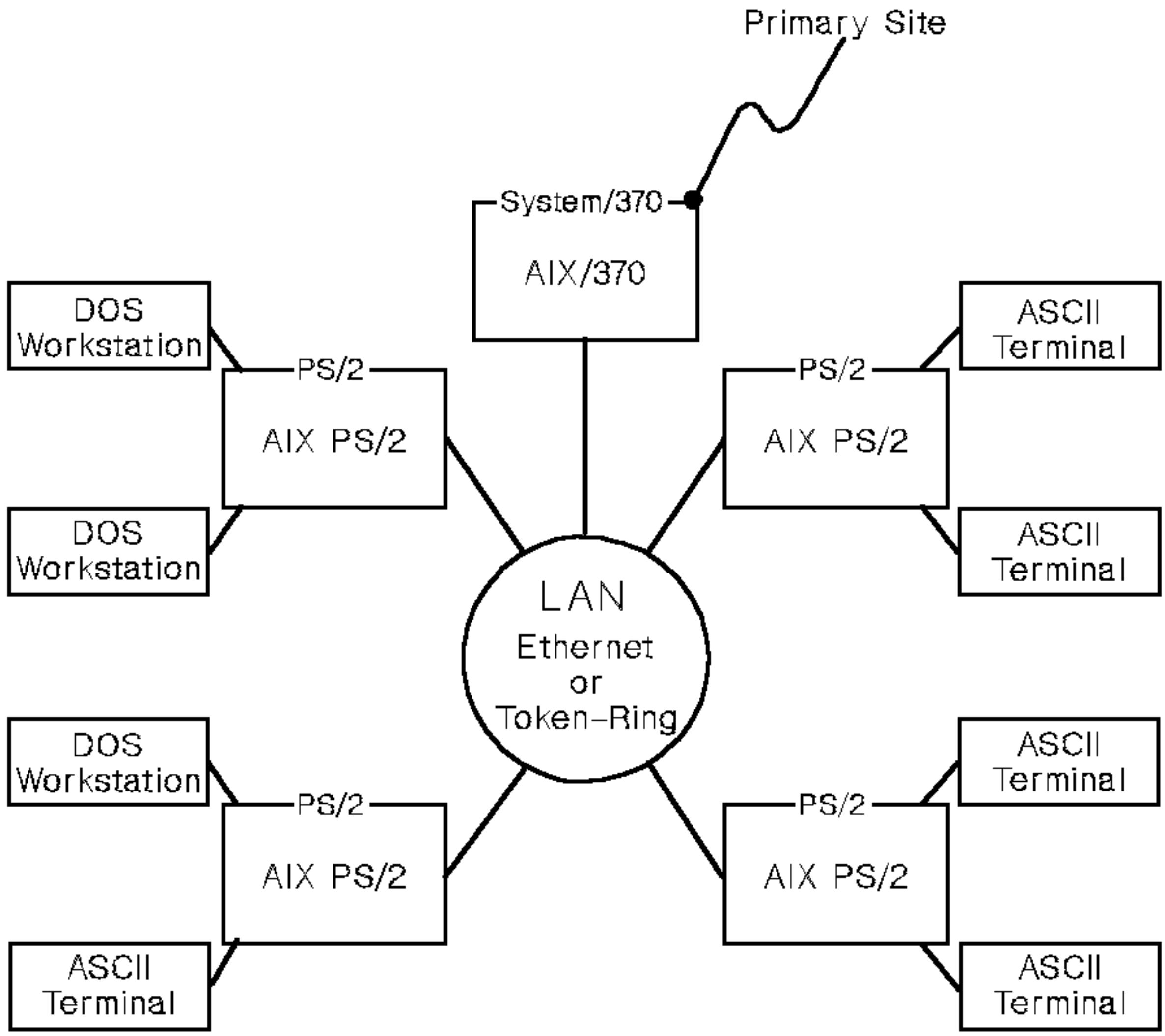


Figure 3-3. Diagram of Heterogeneous TCF Cluster

3.3.3.4 The RT in the Cluster

The IBM RT computer (6150) uses the Reduced Instruction Set Computer (RISC) processor for speed.

AIX/RT does not include TCF functional ability. The AIX/RT machine can run as a standalone main server which supports accessory terminals but it cannot act as a host in a TCF cluster. It makes an excellent programmable workstation, however, and can participate in a cluster in that role.

3.3.4 Interconnecting Clusters

Separate TCF clusters may be joined so that data can be exchanged easily. This requires that one machine be selected as the ***gateway*** through which all such data will pass. The two clusters may use different types of LAN (that is, an Ethernet LAN cluster may be connected to a Token-Ring LAN cluster).

A PS/2 Model 80 running AIX PS/2 may be used as the gateway in some cases. If network traffic between the clusters is extensive, however, a System/370 processor is recommended to prevent unacceptable propagation delays. TCF requires prompt IP response time. Data transmission times of greater than 300 milliseconds may cause timeouts.

Clusters that are separated by large distances may be joined by X.25 software.

Figure 3-4 illustrates two TCF clusters connected by a System/370 gateway. As shown, these machines may all be tied into a single cluster, but TCF has a limit of 31 sites. This configuration will most often be used when the total number of hosts exceeds that limit. Thirty-two or more hosts can be divided into two clusters and linked as shown.

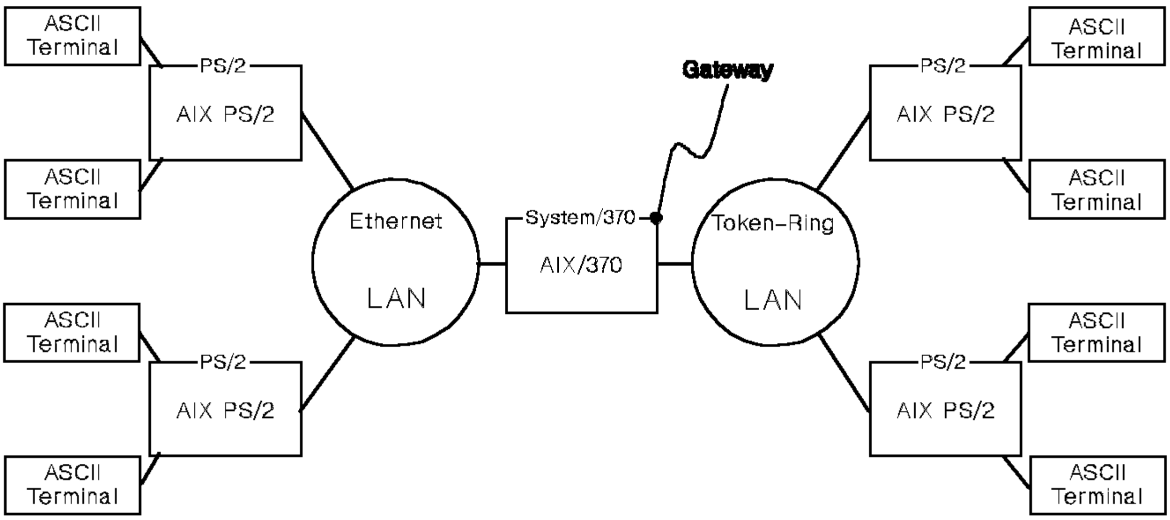


Figure 3-4. An AIX/370 gateway is used to connect an Ethernet LAN cluster to a Token-Ring LAN cluster.

3.3.5 Programmable Versus Fixed-Function Terminals

Programmable terminals are those with enough processing ability to share some of the processing burdens with the host. They usually have memory, but no disk or tape storage. They often have the ability to detect whether their transmissions have been properly received by the intended recipient and to repeat those message packets which went astray. Some programmable terminals may have special enhancements, such as display features (underline, reverse video boldface, blinking characters, and so on) and built-in communications protocols.

By contrast, a fixed-function terminal is a simple display terminal which depends entirely on the host for all its processing. It can send and receive data as a simple ASCII data stream, and nothing more. It transmits and receives data, one character at a time, with start and stop signals at the beginning (and end) of each.

The PS/2 is equipped with an asynchronous adapter; the System/370 is not. Therefore, AIX PS/2 can handle the asynchronous RS-232 data transmissions used by fixed-function terminals while AIX/370 can only receive transmissions in TCP/IP format. Therefore, the terminals in an AIX/370 network must either be capable of handling TCP/IP protocols themselves, or their data must be transformed into that format.

The fixed-function or ASCII terminals used in such a network must send their data to a PS/2, which will transform it into TCP/IP packets and pass it along to the System/370 machines.

Figure 3-5 shows two methods of connecting user display stations to the AIX/370 Operating System.

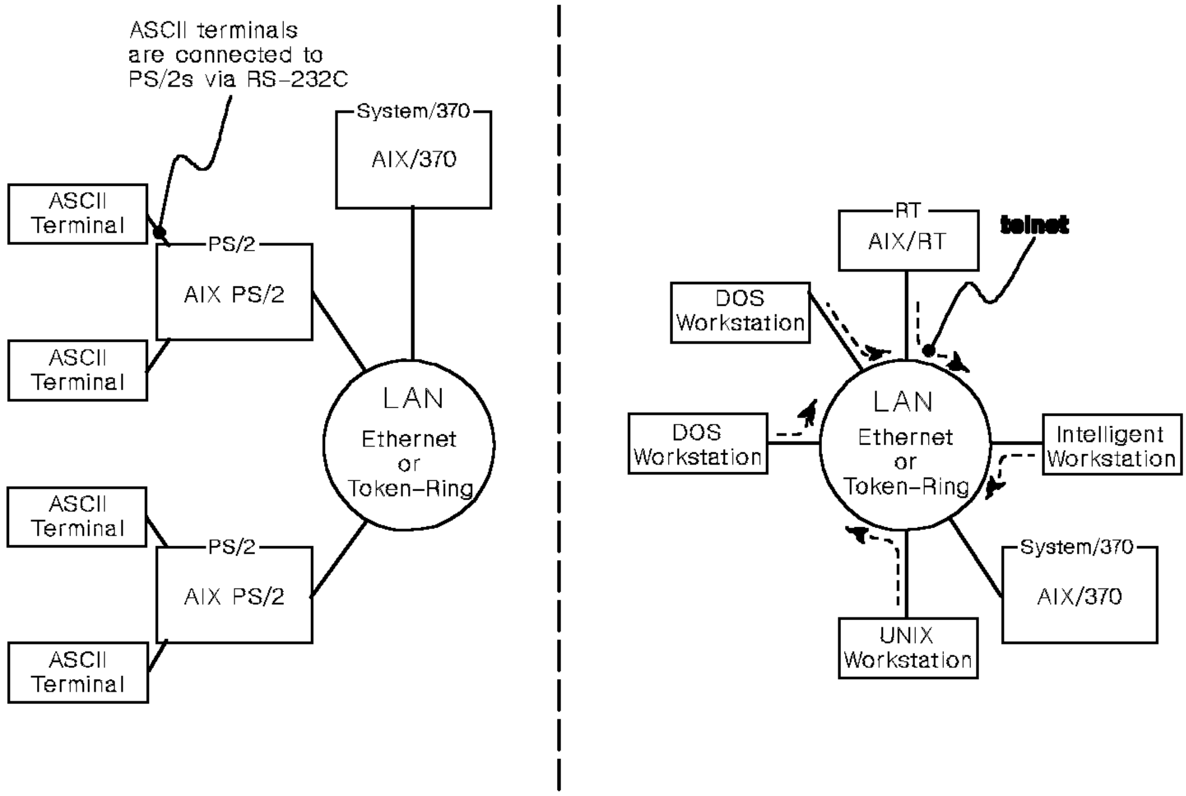


Figure 3-5. Two Methods of Connecting User Display Stations to AIX/370

3.3.6 Attaching Fixed-Function Terminals to AIX/370

The following methods allow ASCII terminals to be connected to AIX/370:

Terminal to AIX PS/2 to AIX/37

Terminal to workstation to AIX PS/2 to AIX/37

Terminal to workstation to AIX/370

Subtopics

3.3.6.1 Terminal to AIX PS/2 to AIX/370

ASCII terminals can be connected to a PS/2 running AIX. The PS/2 can then pass data through to an AIX/370 machine which is part of its own cluster. The RS-232 asynchronous transmissions from terminals are transformed into TCP/IP format and passed along over the LAN.

Figure 3-6 shows how to use an AIX PS/2 to connect ASCII terminals to AIX/370.

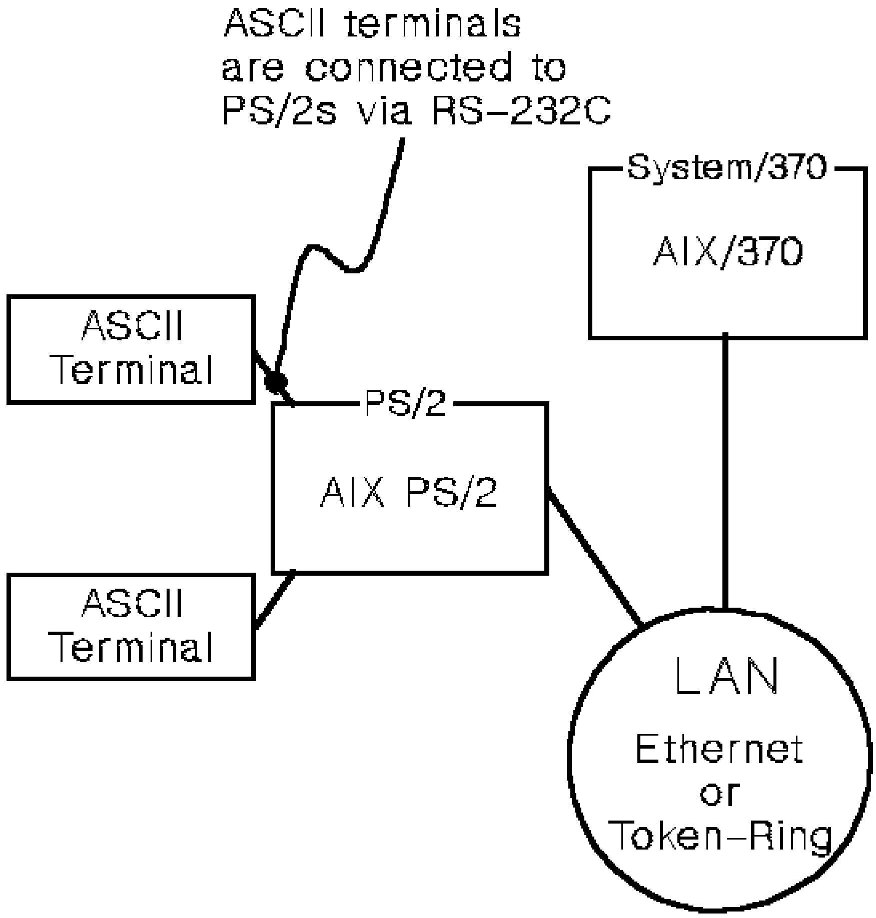


Figure 3-6. Using AIX PS/2 to Connect ASCII Terminals to AIX/370

The PS/2 can handle up to 31 ASCII terminals concurrently. This is the preferred attachment method because it offloads full duplex character I/O from the AIX/370 machines of the cluster onto the AIX PS/2 machines. With TCF running, the users see a single system image and have access to all the resources of the cluster. Highly interactive processes will run directly on the PS/2 while CPU-intensive processes will be automatically and transparently executed on the System/370.

The preferred method of attaching ASCII terminals to PS/2s is to use the multiport RS-232C adapter cards in the PS/2s. This offloads some of the character I/O processing from the 80386 central processor onto the adapter itself. The result is better response time for the user and decreased load on the PS/2.

3.3.6.2 Terminal to Workstation to AIX PS/2 to AIX/370

ASCII terminals may be connected to either a RT running AIX/RT or a UNIX workstation. The RT or the UNIX workstation uses TCP/IP to **telnet** to the AIX/370 machine via a PS/2 which is part of the same cluster. The PS/2 must be running AIX PS/2, and will pass the TCP/IP **telnet** traffic through its LAN adapter board rather than the RS-232C asynchronous adapter.

Figure 3-7 shows how to use programmable workstations and AIX PS/2s to attach ASCII terminals to the AIX/370 Operating System. This option requires the definition of pseudo-**tty** devices on the PS/2. Up to 31 users may connect with each PS/2.

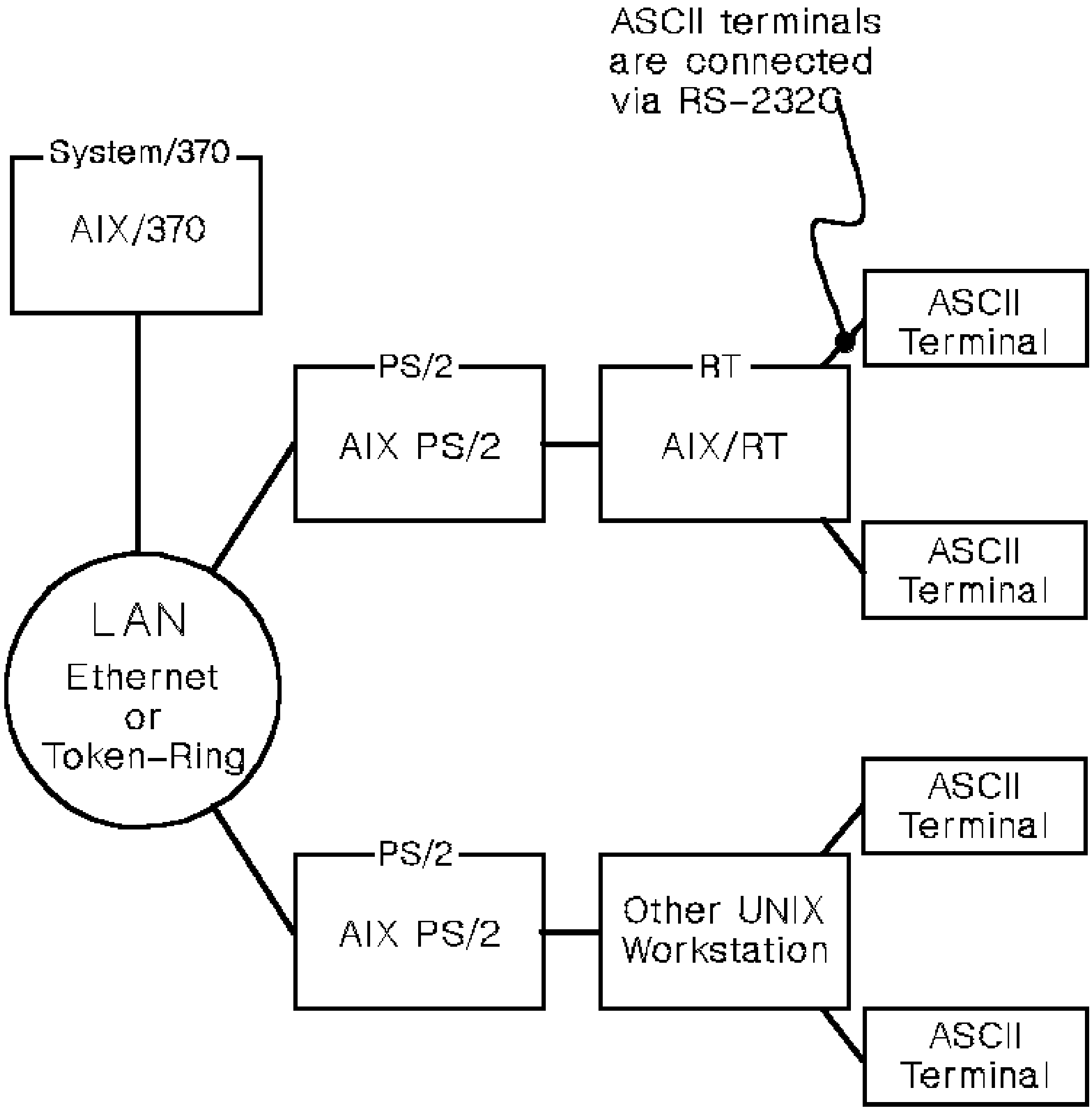


Figure 3-7. Programmable workstations and AIX PS/2 attach ASCII terminals to AIX/370.

3.3.6.3 Terminal to Workstation to AIX/370

ASCII terminals can be connected to a RT or UNIX workstation. The AIX/RT or UNIX workstation can **telnet** directly over the LAN to the AIX/370 machine.

Figure 3-8 shows how programmable workstations connect ASCII terminals to the AIX/370 Operating System.

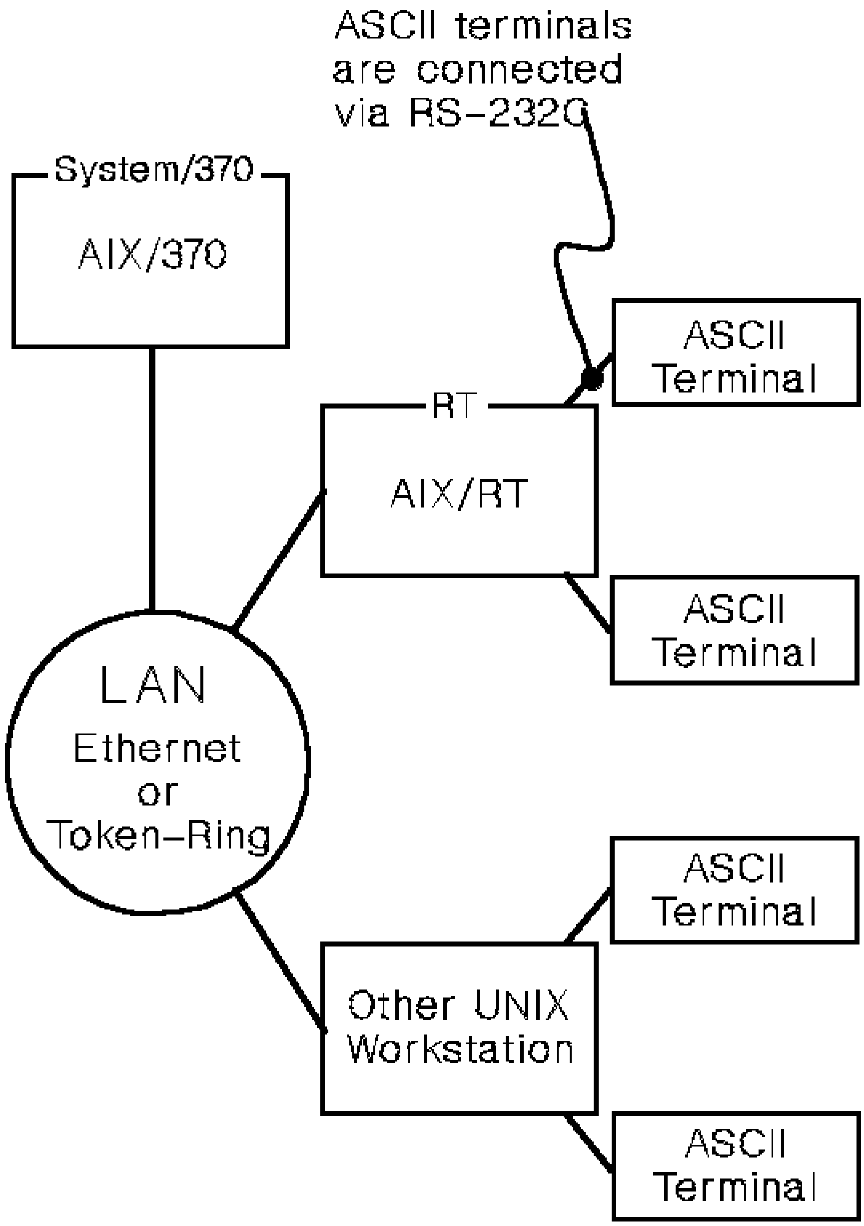


Figure 3-8. Programmable workstations connect ASCII terminals to AIX/370.

This method should be employed only for occasional usage of short duration. Only small files should be processed. The advantage of the previous two methods is lost when the PS/2 is removed from the line. Characters may have to be processed exclusively by AIX/370 and transferred across the LAN, one character at a time. Performance in this situation is slow for prolonged or frequent use.

3.3.7 Attaching Programmable Workstations to AIX/370

The following methods allow programmable workstations to make connection with AIX/370:

AIX Access for DOS Users to AIX PS/2 to AIX/37
AIX Access for DOS Users via LAN to AIX/37
X-Windows for DOS to AIX/37
TCP/IP for DOS to AIX/37
NFS File Sharing and Remote Process Execution

Subtopics

3.3.7.1 AIX Access for DOS Users to AIX PS/2 to AIX/370

PCs or PS/2s can run AIX Access for DOS Users software. The machine running AIX Access for DOS Users acts as the programmable workstation. It must connect with a PS/2, which is part of the desired cluster, and which is running a complementary software program called DOS Server.

Figure 3-9 shows how AIX Access for DOS Users can connect with AIX/370 (via AIX PS/2), either through direct RS-232 lines or through **telnet** (over the LAN).

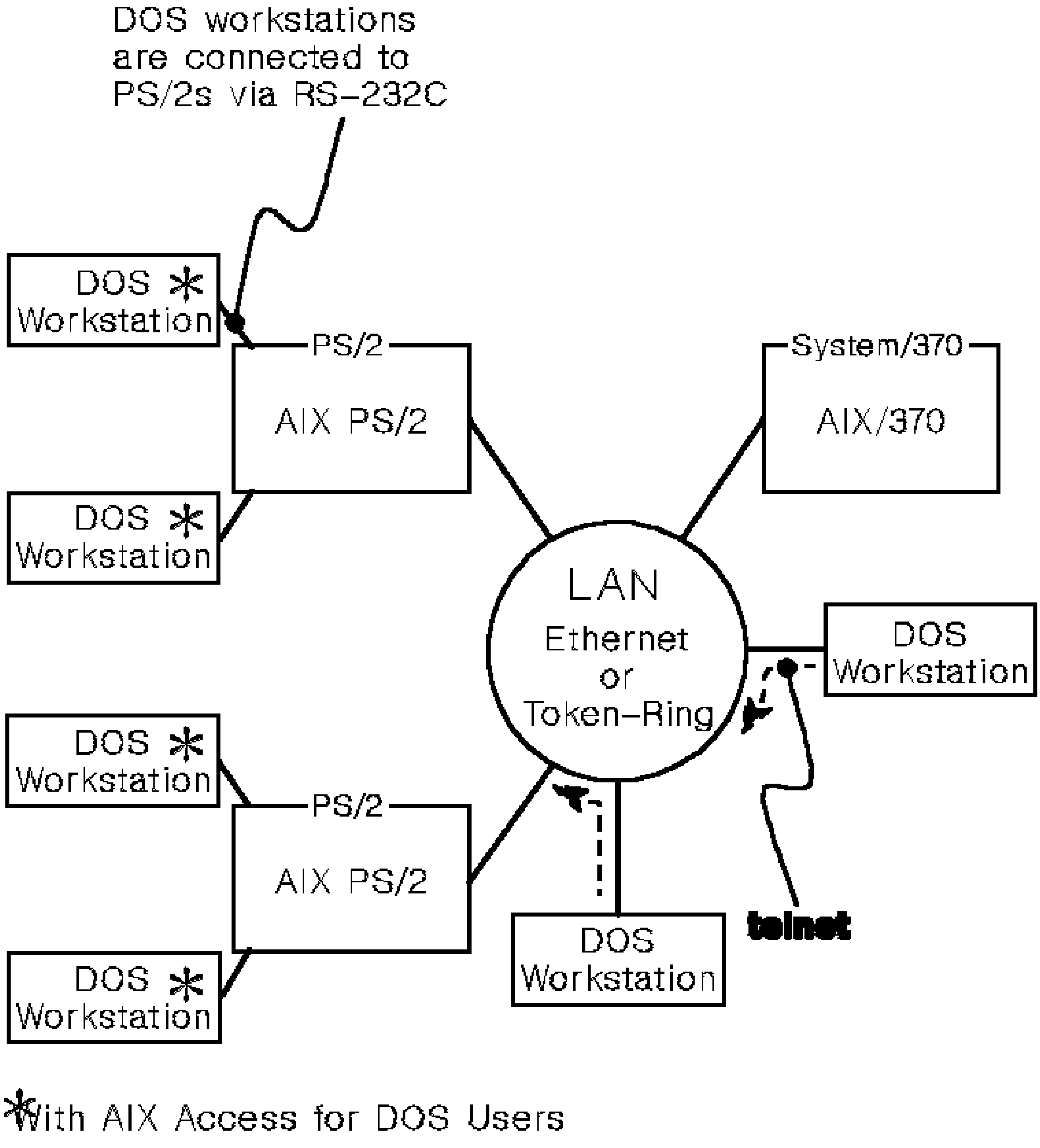


Figure 3-9. AIX Access for DOS Users can connect with AIX/370 (via AIX PS/2), either by direct RS-232 lines or by telnet (over the LAN).

The DOS machine (either PC or PS/2) emulates an ASCII terminal through software. The connection with the AIX PS/2 host may then be made by any of the following:

Ethernet LA
Token-Ring LA
RS-232C line

The PS/2 then passes data along to the AIX/370 machine over the LAN.

This option is preferred for the same reasons as those stated for real ASCII terminals connected to AIX PS/2. In addition, much of the intensely interactive processing done at any terminal, especially editing, can be offloaded onto the PC itself.

AIX Access for DOS Users offers transparent file sharing of the AIX file system. The AIX file system becomes available from the DOS environment and may be manipulated with ordinary DOS commands. These DOS-to-UNIX file manipulations put most of the processing burden on the CPU of the DOS machine and on the PS/2 running DOS server. Even if the files are stored on an AIX minidisk of the AIX/370 machine, the AIX/370 machine will do only a small amount of the processing.
Even when conducting a terminal emulation session on the AIX network, the AIX Access for DOS Users programmable workstation carries part of the processing load. A function called Remote Control Terminal Echoing (RCTE) provides echoing of characters locally and only sends data to the host in blocks, when a break character similar to the **Enter** key is pressed. This facility does not work when running raw mode AIX applications like the vi editor. For this reason, a DOS version of vi is provided. This offloads the intensely interactive editing function onto the DOS CPU.

3.3.7.2 AIX Access for DOS Users via LAN to AIX/370

A PC or PS/2 running AIX Access for DOS Users can use the terminal emulation capabilities of AIX Access for DOS Users to conduct a standard UNIX session on the AIX system. All I/O goes directly across the LAN to the desired host.

Figure 3-10 shows how a PC or a PS/2 can use the AIX Access for DOS Users **telnet** feature to connect with the AIX/370 Operating System.

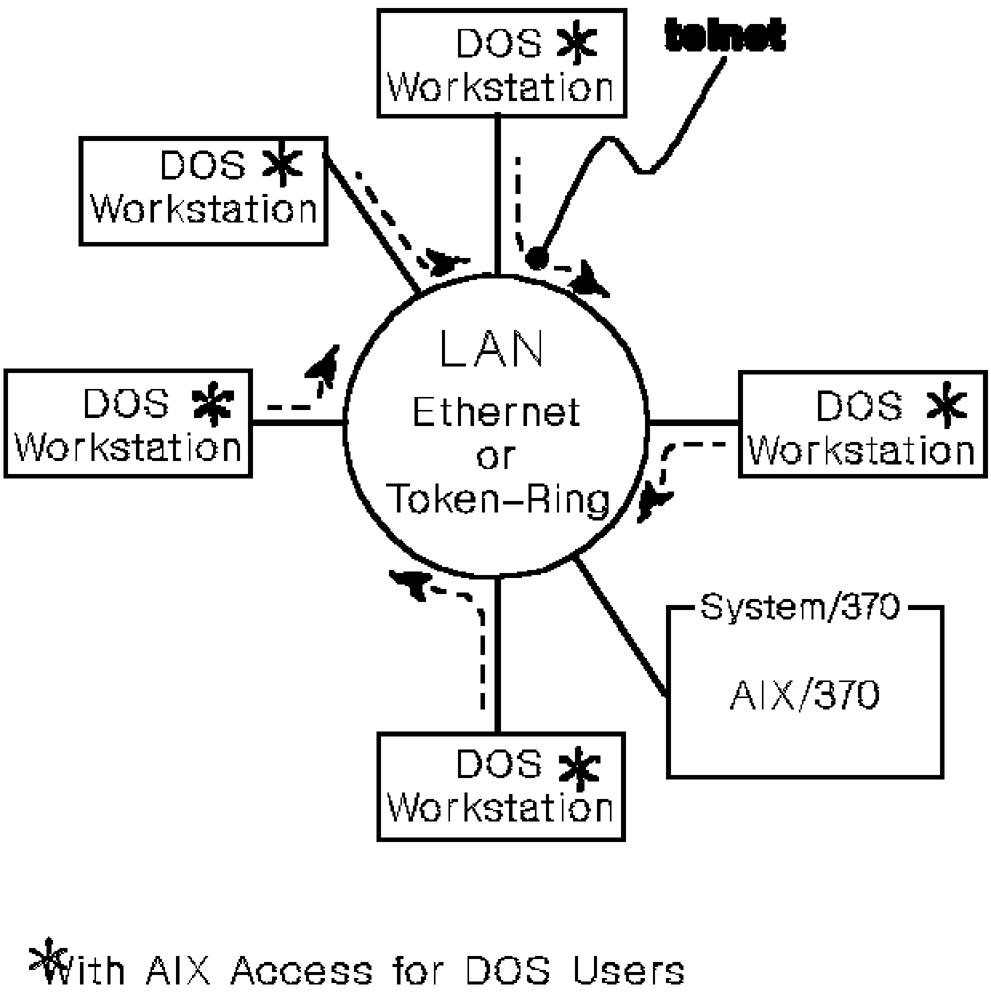


Figure 3-10. PCs or PS/2s can use the AIX Access for DOS Users telnet feature to connect with AIX/370.

Since this option removes the PS/2 from the data path, its terminal concentrator function is lost. This places slightly more load on the AIX/370 host than the previous method. Nevertheless, this method still retains the advantages of RCTE character processing in DOS, and the DOS form of the vi editor can still be used to offload editing.

3.3.7.3 X-Windows for DOS to AIX/370

X-Windows for DOS executes on either a PC or a PS/2. The machine must be equipped with a graphics card and a bit-mapped display. All communications flow directly across the LAN from the programmable workstation to the AIX/370 machine and back.

Figure 3-11 shows how X-Windows for DOS permits a DOS machine to make contact with an AIX/370 machine directly across the LAN.

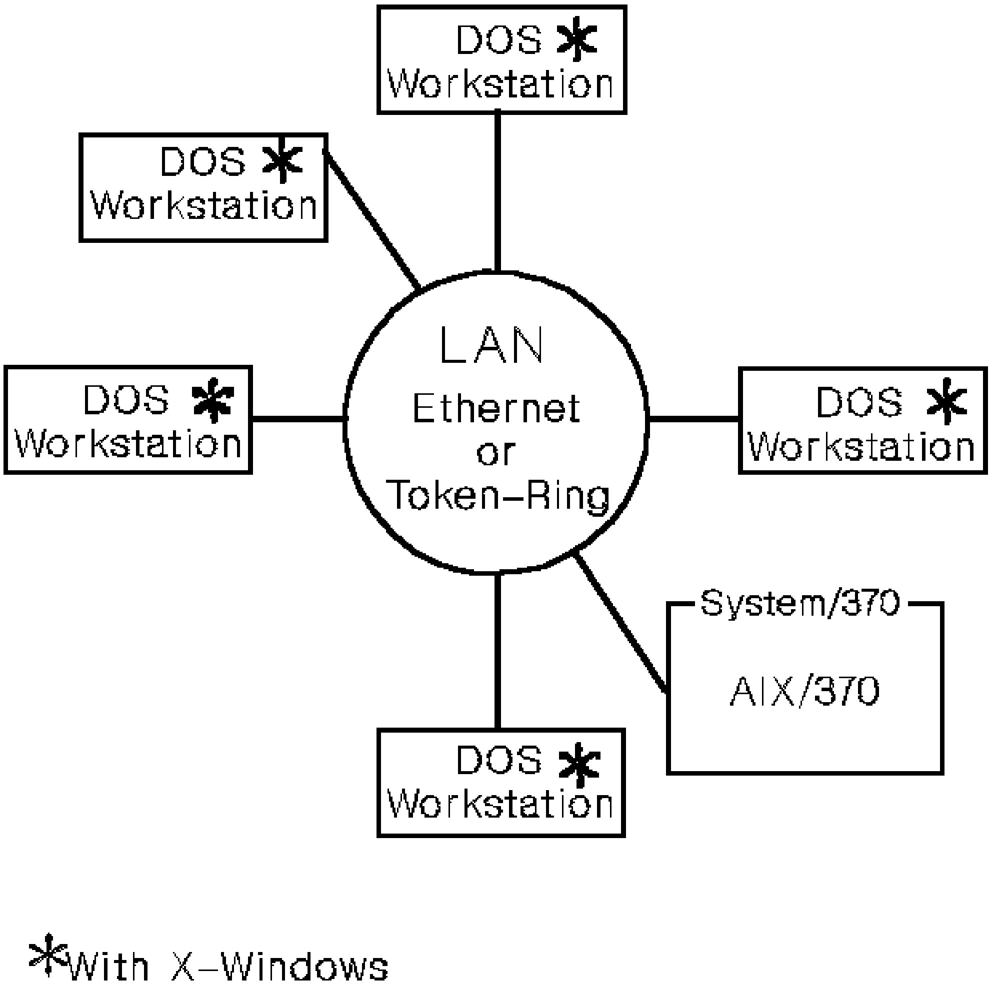


Figure 3-11. X-windows for DOS permits a DOS machine to make contact with AIX/370 directly over the LAN.

The work of managing screen graphics is offloaded from the host onto the workstation. An application running on the AIX/370 machine will act as the X-Windows client. The programmable workstation will receive data across the LAN and its X-Windows Server software will process the data into a screen image.

3.3.7.4 TCP/IP for DOS to AIX/370

Either the PC, PS/2, or VM TCP/IP (which then passes the communication along to AIX/370) may execute TCP/IP. This software communicates directly across the LAN to the TCP/IP for VM which is executing on the AIX/370 machine. The programmable workstation uses **telnet** to access the AIX/370 host across the LAN. Here again, the preferred way to use this function would be to **telnet** to a PS/2 which is running TCF and is part of the cluster with the desired AIX/370 machine.

Figure 3-12 shows how TCP/IP for DOS allows DOS workstations to make direct contact with the AIX/370 Operating System across the LAN.

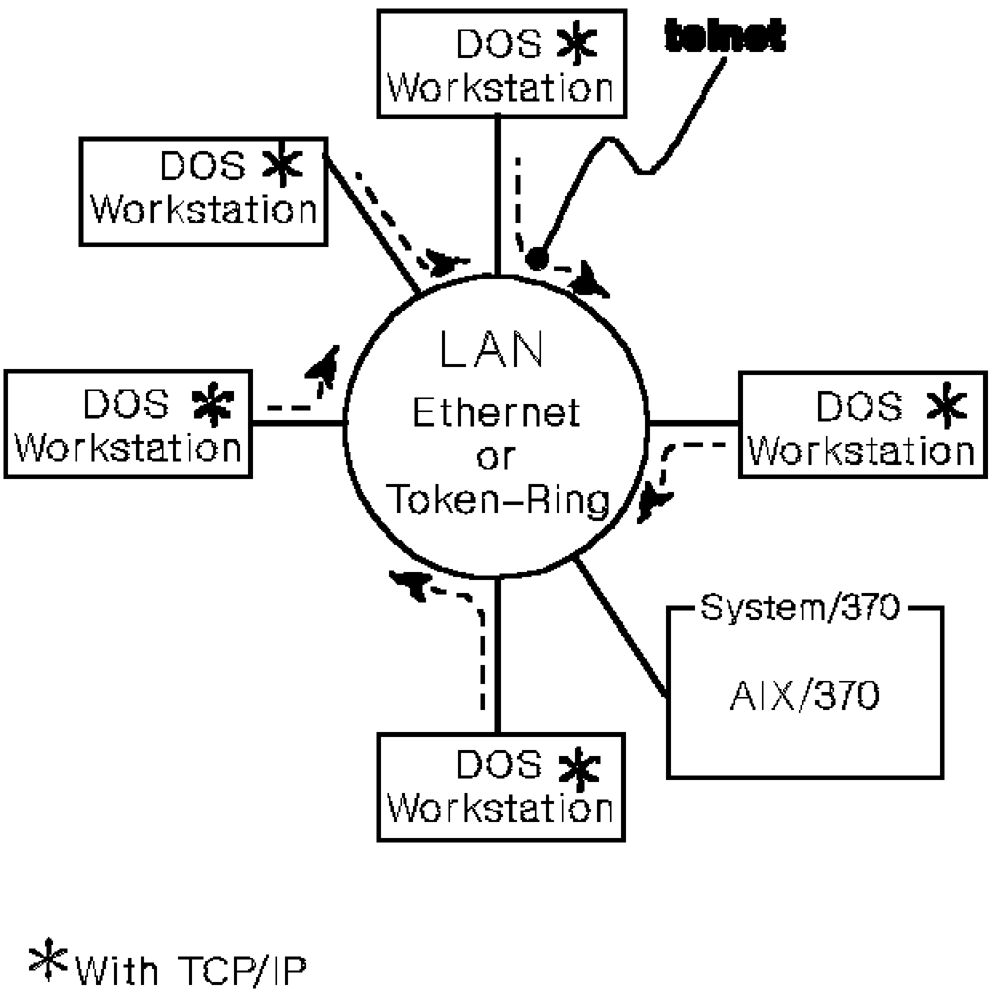


Figure 3-12. TCP/IP for DOS allows PCs or PS/2s direct contact with AIX/370 across the LAN.

This choice is not as desirable as the AIX Access for DOS Users option since RCTE is not used to enhance performance. In addition, the escape sequences used by various terminals are not well supported, so UNIX applications that use control sequences to control cursor movement will not work. This includes the vi and INed editors.

3.3.7.5 NFS File Sharing and Remote Process Execution

An AIX/RT or UNIX machine can use NFS to access file systems located on the AIX/370 machine. Processes can be started on the System/370 by either of two methods:

TCP/IP rexe

NFS RPC (Remote Procedure Call)

Figure 3-13 shows how NFS can provide file sharing and limited processing ability for programmable workstations.

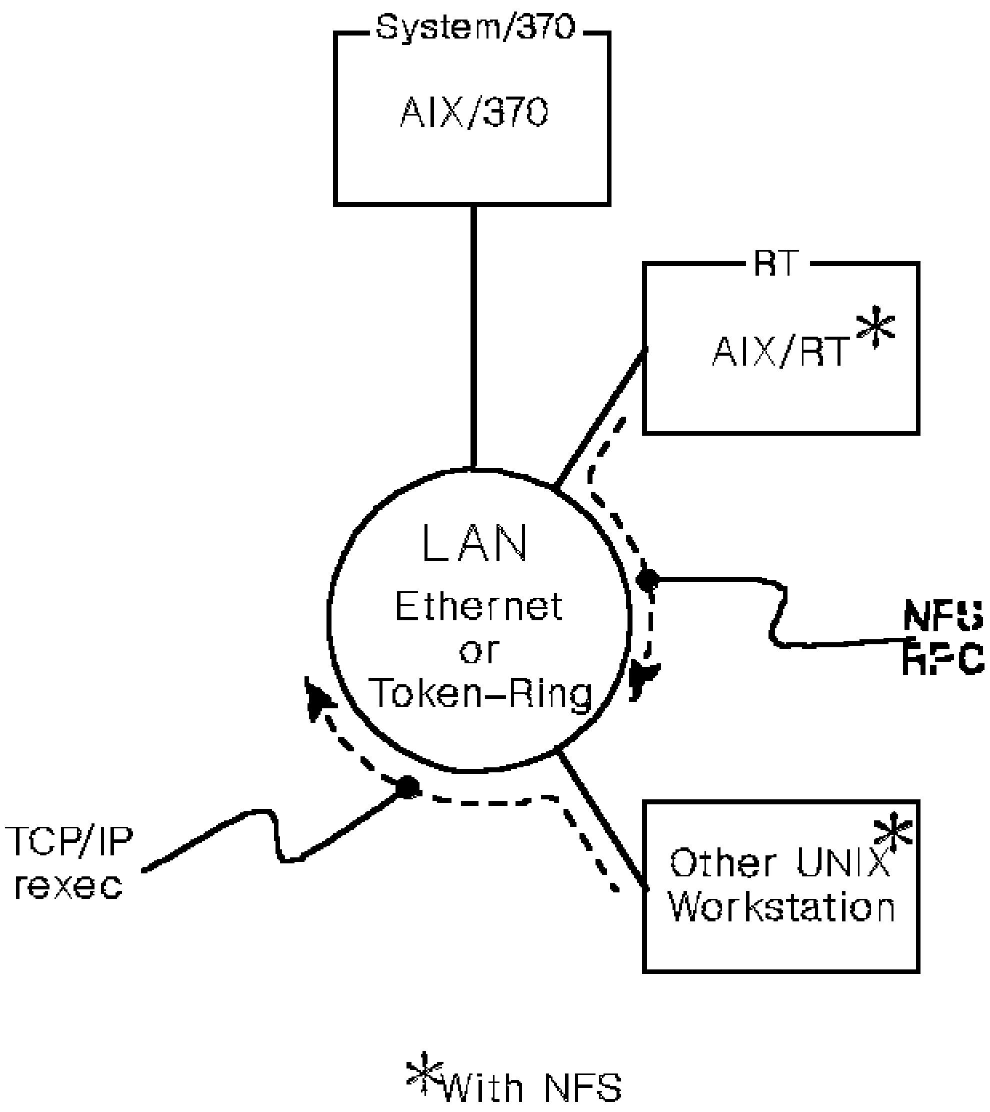


Figure 3-13. NFS provides file sharing and limited processing ability for programmable workstations.

For more information on the **rexec** function, refer to the *AIX Operating System TCP/IP User's Guide*. For further information on the NFS facilities, refer to *Managing the AIX Operating System*.

This option does not allow ASCII terminal support or transparent execution of commands on the cluster; remote execution commands are used for these applications. It does, however, allow the application to have transparent access to file systems on the AIX/370 machine. This may prove sufficient for some applications.

3.3.8 Terminal Arrangements in the Cluster

The following sections describe terminal arrangements in the homogeneous and heterogeneous clusters.

Subtopics

3.3.8.1 Terminal Arrangement in Homogeneous Clusters

In any of the homogeneous clusters, there is no particular reason to prefer attachment of terminals to one host over another. The only consideration here is to keep the terminals fairly evenly arrayed around the cluster. Ideally, each cluster host will support approximately the same number of terminals.

Figure 3-14 shows how user terminals should be evenly distributed in a homogeneous cluster.

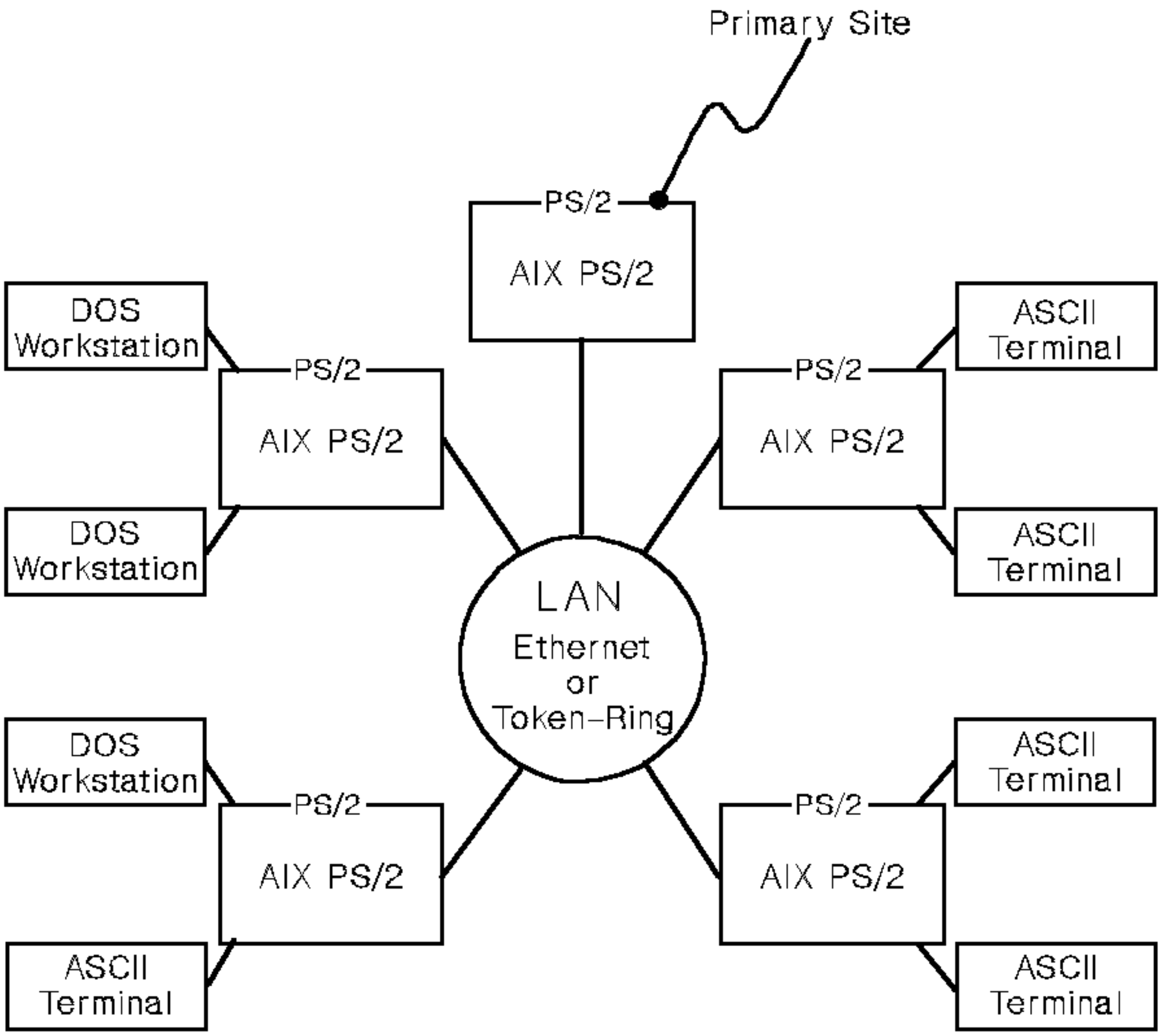


Figure 3-14. User terminals can be evenly distributed in a homogeneous cluster.

The sole exception to this rule is to keep the primary site lightly loaded as compared to the other sites. This makes the primary site the least likely to be overloaded by interactive user requests. Follow the rules given in "Cluster Balancing" in topic 3.4.

3.3.8.2 Terminal Arrangement in Heterogeneous Clusters

In a heterogeneous cluster, all fixed-function terminals are connected to the AIX PS/2 machines of the cluster rather than the AIX/370 machines. This has the effect of offloading most of the character I/O from the AIX/370 machines onto the PS/2s. Only programmable workstations are connected to the LAN.

Figure 3-15 shows how the user display stations in a heterogeneous cluster should be connected to the PS/2s. Programmable workstations may also be connected to the LAN.

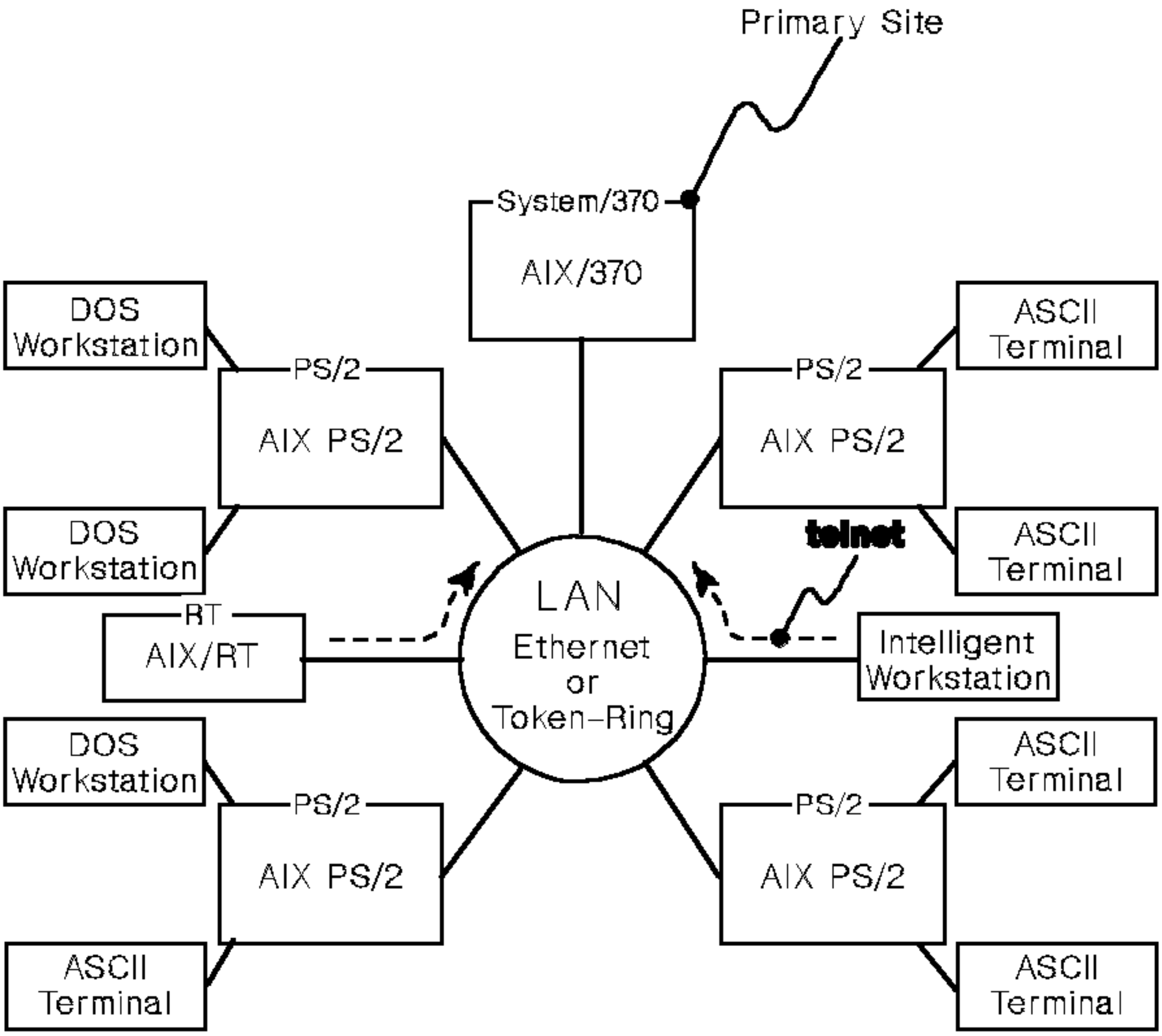


Figure 3-15. Diagram Showing How User Terminals in a Heterogeneous Cluster Should Be Connected

Mainframes are most efficient at handling large, CPU-intensive jobs. PS/2s are far better suited to jobs that are intensely interactive and require a great deal of character I/O (refer to "Balancing Editor Usage" in topic 3.4.3).

3.3.8.3 Decisions to Make about Terminal Usage

In planning for the terminals to install on your AIX/370 system, decide:

How many users need to be served

How many of these will need their own, separate terminals o workstations? If your organization works in shifts or if certain individuals need terminal access only rarely, some display stations may be shared.

How many display stations your organization already has

How many of those can be adapted for use with AIX/370

Valid workstations can be any of the following:

PS/2s or RT

DOS machine

ASCII terminal

Any UNIX workstatio

Any machine capable of emulating a VT100 termina

Any machine capable of running TCP/IP, or X-Windows

For each PS/2 which is to act as a terminal concentrator in the cluster, decide:

How many ASCII terminals it will service

How many programmable workstations it will service

What software packages those workstations will run

3.4 Cluster Balancing

The cluster must be kept balanced in two senses: processor workload balancing and file access balancing.

Processor workload balancing means keeping the average workload of each host within the processing ability of that host. If the cluster is homogeneous, all hosts should share the total cluster workload in approximately even measures. If different processor types are linked into a heterogeneous cluster, the workload must be apportioned according to processor ability.

File access balancing means keeping network traffic to an acceptable level by storing programs and data files on those hosts where they will most often be used.

Subtopics

3.4.1 File Access Balancing

TCF allows virtually unlimited access to all files in the cluster by any user or program. This is convenient for users who can run any program (for which they have appropriate permissions) against any data file (for which they have the needed permission). The user need not know, or care, which host actually stores these files.

Note: The planner and the system administrator however, should take care to set up the system so that frequently used files and programs are physically stored on the hosts where they will most often be used.

If the user logs onto host A and then runs a program only kept on host B against a data file only stored on host C, every single I/O operation the program performs must take place across the network lines. These LAN channels are relatively slow compared to disk access times. This gives poor performance from the user's viewpoint, and greatly increases the chances that the LAN channel will be overloaded. When the network is completely occupied with sending data back and forth across the LAN connections, relatively few CPU cycles are left for useful work. Overall performance goes down for all users.

If users are separated by tasks (refer to Figure 3-16), they can log on to whatever host stores the programs and data files. Separating users by task is particularly important in dealing with large files.

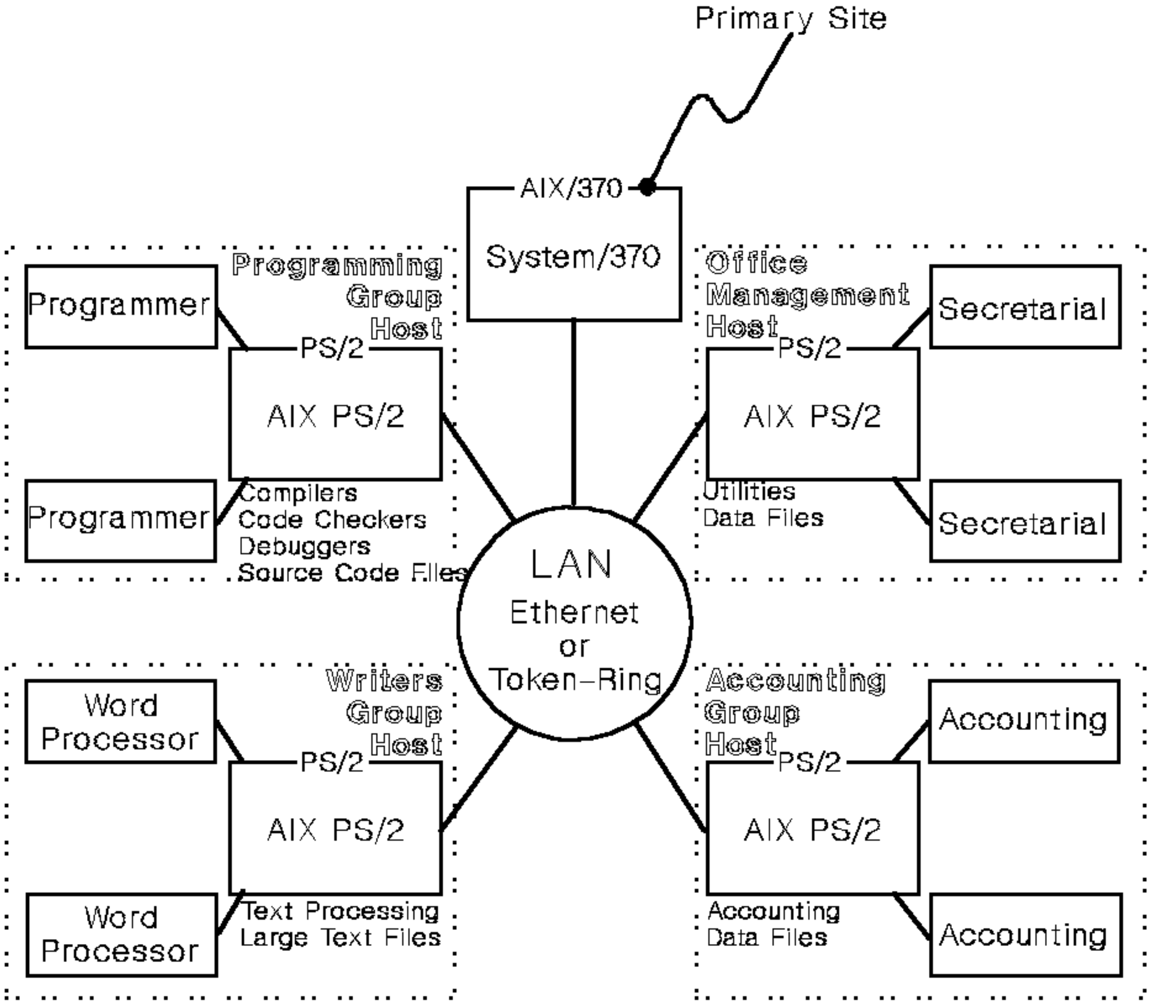


Figure 3-16. Users can be divided into groups according to tasks.

Users who consistently access only small files can log onto hosts other than those where the files are stored (if network crowding or the physical layout absolutely demand this). Overhead in such transfers will be relatively low, and response times will be acceptable.

Users who consistently access files in excess of one megabyte, however, should be placed close to those files, even if physical movement of the users or the network hardware is necessary to accomplish this.

Within any single TCF cluster this is easily accomplished by having users **telnet** to the host where the files are stored, even if physical hardware limitations require them to logon elsewhere. They can also use the **onhost** facility to run their processes wherever the programs or the data files are stored.

In the case of two separate TCF clusters linked by network connections, however, the situation may be more complex. If high-volume users log onto one cluster and the files or programs they need are on the other cluster, a software solution may not be available. If the connection between the two involves **uucp** facilities, performance will be inadequate and network traffic will be unacceptable. In this case, physically moving users, (or switching them to the desired network) should be considered.

3.4.2 Processor Workload Balancing

TCF does not automatically do load leveling by running processes on the site with the lightest load. It does, however, provide some very useful tools for load leveling. A planner or system administrator can use these tools to develop procedures to automatically balance the cluster workload. Such procedures are normally written as AIX shell scripts.

The following list of commands are available in TCF to check processor loads and to run a process on the most appropriate processor.

fast alias Allows the user to specify that a process is to be executed on whichever of the available hosts will run it in the least time.

fastsite Displays the name of the most lightly loaded processor.

loads Displays a list of cluster sites with load averages for each.

migrate Moves a process from one System/370 site to another (only available in AIX/370).

on Allows users to indicate the processor on which they want their process to execute.

3.4.3 Balancing Editor Usage

Full-screen editors, like the standard AIX editor vi, have a significant influence on processor utilization. The process is inherently interactive and CPU intensive. The largest overhead is in refreshing the screen constantly to show changes as they are made. Therefore full-screen editing should be done directly on the System/370 processor with discretion only.

As a general rule, offload editing tasks to some smaller processor, whenever possible. This can be accomplished by running the editor on one of the PS/2 hosts of the cluster or a programmable workstation.

Subtopics

3.4.3.1 Offloading Editing to a PS/2 Cluster Host

When the cluster includes PS/2s running AIX PS/2, the user has the choice of logging onto one of these PS/2 hosts rather than the AIX/370 machine. Users working from an ASCII terminal will normally be connected directly to one of the PS/2s of the cluster and should be encouraged to do their full screen editing there, (when editing small-to-medium sized files). Editing very large files may present a problem, however, and is addressed in "Editing Large Files" in topic 3.4.3.3.

AIX PS/2 supports a vi editor with exactly the same properties as the one which runs on AIX/370. The user will notice no difference when using the two editors.

3.4.3.2 Offloading Editing to a DOS Machine

It is almost always an advantage to use a programmable workstation for editing rather than editing on the network. If this facility is available, the user should be encouraged to use it for nearly all editing. Offloading editing chores to a programmable workstation results in better response time for the user and a lighter load for the CPUs of the cluster.

Interactive editing involves a great deal of load on the CPU. PCs were developed with an emphasis on exactly this sort of intensive user interaction. Mainframes were developed in an atmosphere that stressed batch processing of very long files. Therefore PCs excel at processing small files interactively, while mainframes are best suited for long, CPU-intensive jobs run against large files.

A PC with AIX Access for DOS Users software makes an excellent programmable workstation. The package includes a DOS-based vi editor with full functional ability. AIX editing can therefore be offloaded from the network without the necessity of AIX users learning new editing skills.

Any other DOS editor that the user prefers should work just as well as vi, as long as it is one which produces standard ASCII text.

The DOS editor can even be used to access text files which remain on the network and then edit them in place. This editing is done under DOS, and the files appear to be located on the DOS system. This practice is not advisable as a rule, however, because network overhead and system response time will increase.

The situation is best handled by the simple expedient of copying the AIX file to the PC's fixed disk, editing the file in DOS, and copying it back to the network. AIX and DOS files differ in their handling of the end-of-line symbol, but AIX Access for DOS Users includes utilities for copying the files back and forth between the two formats. This also results in an automatic backup copy of the file (one on the AIX network and one on the DOS disk). Its main drawback is that the user is off the network during the editing session and therefore unavailable for network communication.

The same is true for a PS/2 machine running in DOS mode. Such a machine could function as an AIX programmable workstation while running in DOS mode, and still have full multi-tasking capability when running OS/2.

3.4.3.3 Editing Large Files

File size is the single most important consideration in the decision of where to edit. When TCF is running in the cluster, all files are accessible as parts of the same global file system. Under most conditions, the user does not even know where the file is stored. In the case of very large text files, however, the user needs to be informed.

The mainframe form of the vi editor is very efficient at editing large files. However, the DOS version cannot handle them.

A large file must be broken into several small files in order to handle it in DOS. DOS editors vary in the maximum file size they can process, but the mainframe is generally better at this sort of task.

3.4.3.4 Location of the Editor and the Data File

One final factor to keep in mind when planning for editing loads on the cluster is the location of the program file versus that of the data file being edited.

The worst possible example is that of a user logged onto host A who runs an editor program which is only stored on host B against a data file stored only on host C. Every single operation in such an editing session is going to take place across network lines, and system overhead is enormous.

This can only happen if users are using unusual editing programs which are stored only on one host of the network. The vi editor is found on every host.

If editors other than vi are going to be in common usage on a TCF cluster, they should be made available everywhere by incorporating them in the appropriate position on the replicated root file. Data files which are likely to be edited by a particular group should be kept on the host where members of that group ordinarily log on.

In those few cases where users must edit a data file stored only on a host other than the one they log onto, they should be encouraged to use **telnet** or **rlogin** to log onto the other host and then perform the editing there.

3.4.3.5 Guidelines for Editing

Users working at the console of an AIX PS/2 cluster host should do al their full screen editing on that machine. The sole exception would be editing a very large file.

Users at ASCII terminals should log onto one of the PS/2 hosts and d their editing there. The same exception applies the first rule.

If non-vi editors are to be in widespread use, make them available o every site.

Store large text files on the AIX/370 machine and encourage users t log on and edit there.

For small-to-medium sized text files, offload editing to DOS as first choice.

Use AIX as storage for the finished files. This allows them to be sent to a wide variety of network printers. Copies of the files should be kept in DOS environment where much of the editing can be performed. Offload editing to AIX PS/2 as a second choice.

3.5 Examples of AIX/370 Configurations

The following sections contain typical AIX/370 configurations.

Subtopics

3.5.1 AIX/370 Single-Site Configuration

Figure 3-17 shows a single-site AIX/370 computer configuration. Depicted is one AIX/370 system connected to a LAN along with:

A PS/2 with the AIX PS/2 Operating System installed and IBM 316 (ASCII) terminals attached to it
An IBM PC/XT or AT running X-Windows for DO
A PS/2 or IBM PC/XT or AT running AIX Access for DOS User
Other host machines with TCP/IP installed. Host machines refer t other AIX clusters or single-sites running separate operating system software.

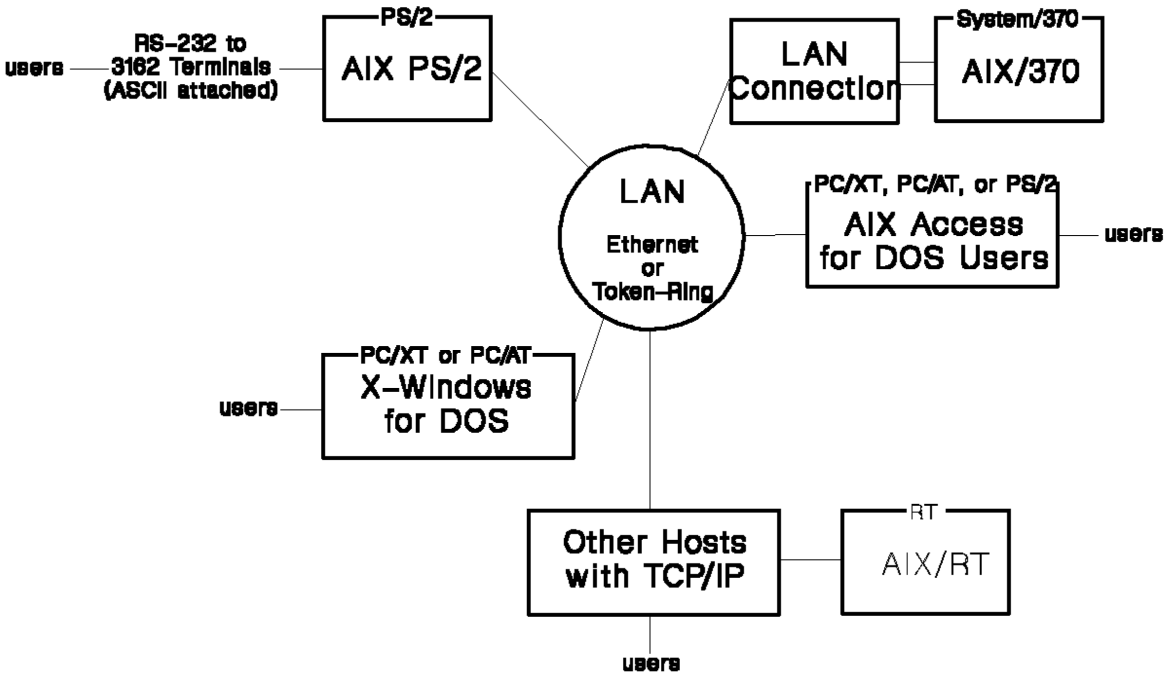


Figure 3-17. Single AIX/370 Configuration

AIX Access for DOS Users allows a user to run DOS applications using data from an AIX/370 (or AIX PS/2 or AIX/RT) file system through an Ethernet, asynchronous (to AIX PS/2 or AIX/RT), or TCA connection. Programs can be downloaded from the AIX/370 system to the PC and executed on the PC. Data files on AIX/370 can be accessed from the PC also.

AIX Access for DOS Users also emulates a VT-100 terminal, allowing the user to conduct a regular AIX/370 session.

X-Windows allows for window access from the PC to the AIX/370 system. The **readme** file (supplied with the X-Windows software) explains how X-Windows works.

Note: X-Windows provides the server function when communicating with AIX/370.

3.5.2 AIX/370 Multi-Site Configuration

Figure 3-18 shows a TCF cluster, which is a group of PS/2 and/or System/370 computers running the AIX Operating System and communicating through a LAN. Ethernet or Token-Ring are two examples of a LAN. AIX/370 systems can also communicate via channel-to-channel and virtual channel-to-channel.

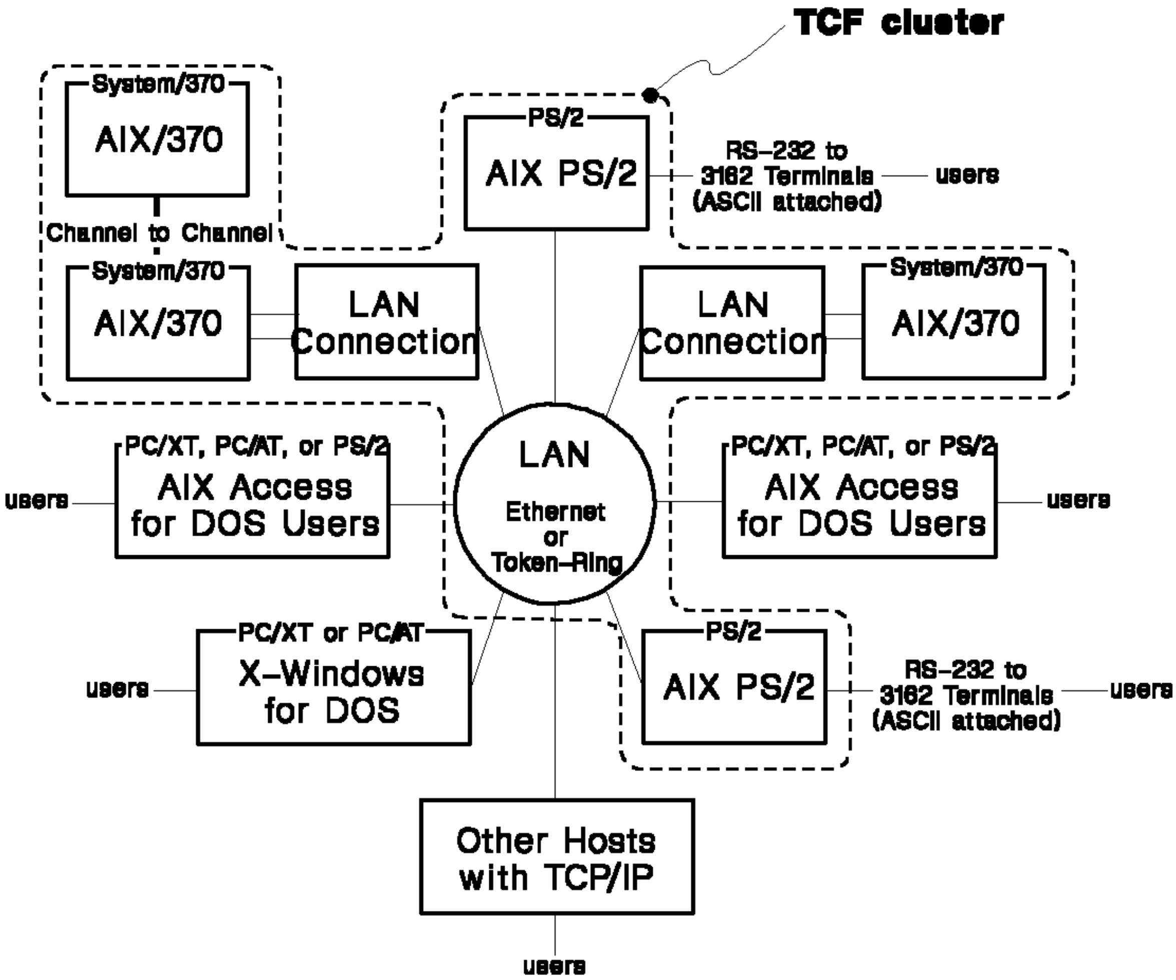


Figure 3-18. Multiple AIX/370 Configuration

Figure 3-18 shows several AIX/370 system machines and AIX PS/2 systems machines that are all part of the same cluster. They are all connected by Ethernet, Token-Ring, or through a channel-to-channel adapter. Note that within the cluster, users gain access over a LAN through consoles and terminals attached to the AIX PS/2 systems. Users also gain access through:

PCs and PS/2s running AIX Access for DOS User
PCs and PS/2s running X-Window
Host computers that connect to a LAN with TCP/IP

3.5.3 AIX/370 Telecommunications

Figure 3-19 shows a multiple computer configuration of an AIX/370 and non-AIX/370 UNIX systems. An AIX/370 system can be connected via a LAN to an AIX PS/2 system which, in turn, can be connected to a wide variety of systems.

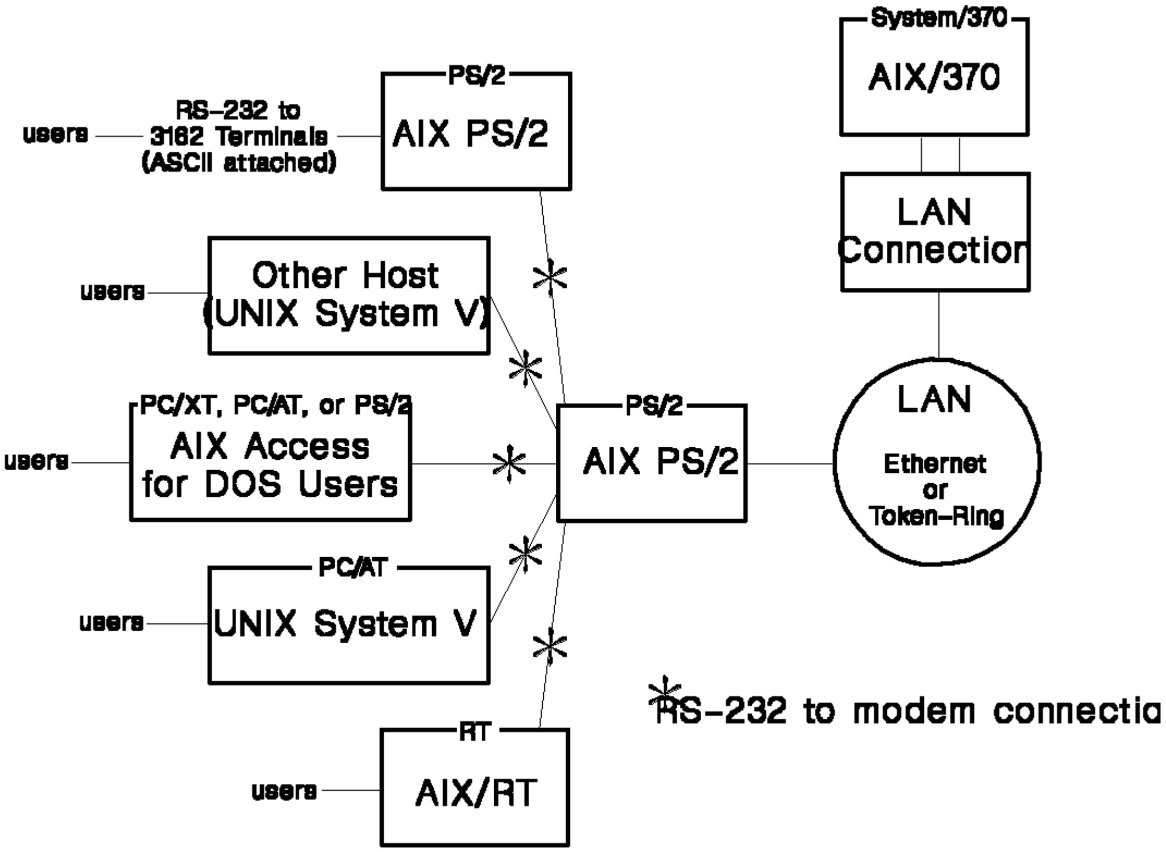


Figure 3-19. Multiple Computer Configuration Teleprocessing

Each computer essentially runs independently from all other computers connected to it. Simple file transfers are possible through the **uucp** (unix-to-unix copy) command over an RS-232 connection. This teleprocessing environment uses the facilities of **uucp**, **uux**, **connect** and **INftp**. It also uses INmail and INnet.

Figure 3-19 illustrates a teleprocessing environment in that the primary mode of communication is through application-level protocols that take place in a point-to-point fashion, in contrast to the cluster environment where the primary communication is based on kernel-to-kernel level protocols operating in a multi-point fashion. Also, the teleprocessing environment uses the RS-232 bandwidth while the cluster environment uses LAN bandwidth.

3.5.4 Combined Configuration

The AIX/370 system allows immense flexibility in terms of hardware and software connections that are available. Any combination of configuration types can be merged into one. For example, all three types of configurations are shown in the sample network configuration in Figure 3-20.

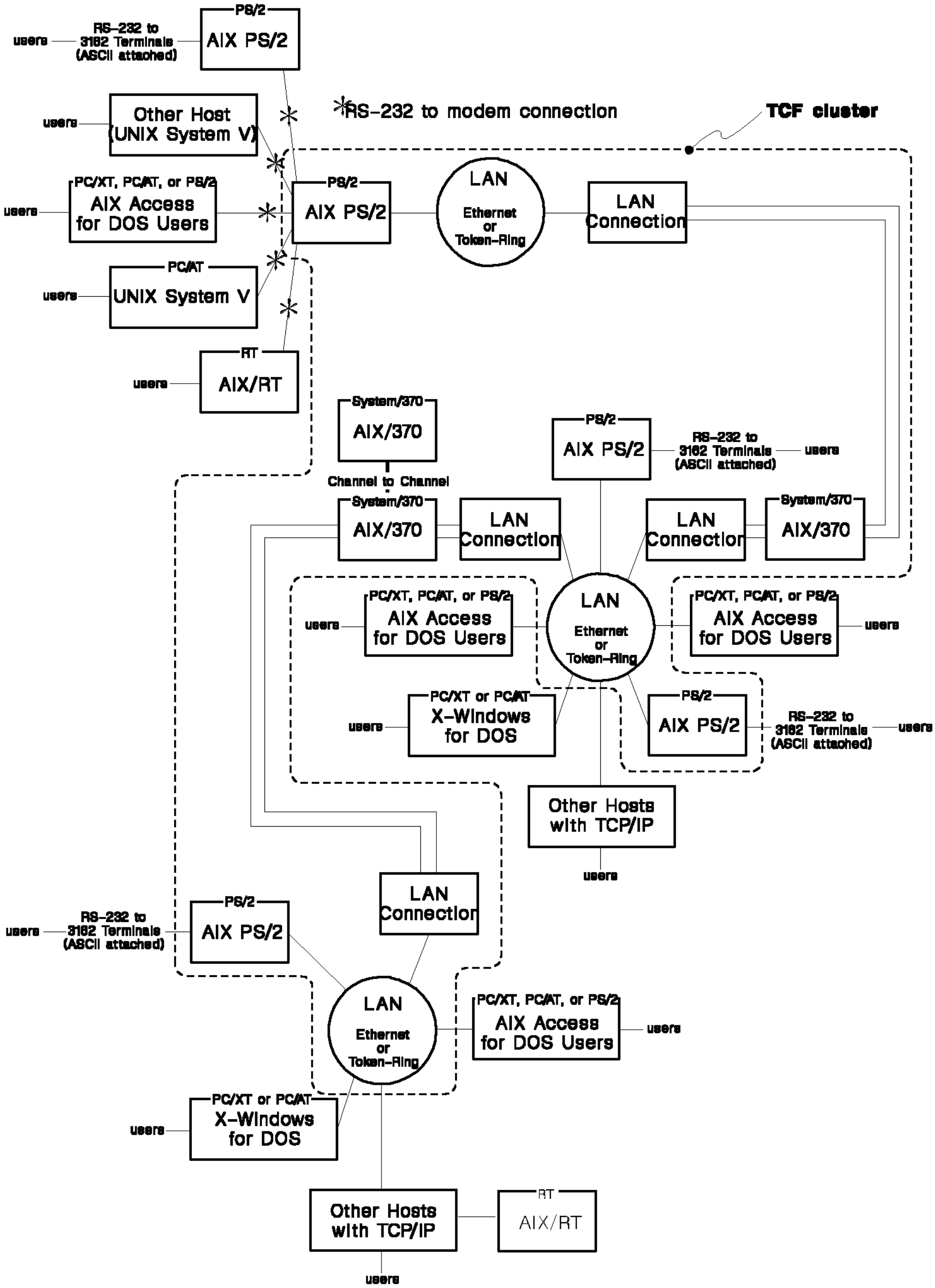


Figure 3-20. Combined Configurations

3.6 TCF Cluster Considerations

Subtopics

3.6.1 LAN Configuration

For configuring LAN devices, the following information is needed:

- The virtual device addresses to be used for 8232, virtua channel-to-channel, 3088, or integrated 9370 adapter
- Selection/assignment of TCP/IP addresses to each LAN interface in th configuration.

It is essential that the LAN be configured to allow any two sites to exchange a packet via a maximum of two gateways (relay sites).

The LAN can often be the shortest route available to provide connection between AIX/370 Operating Systems.

Figure 3-21 shows four AIX/370 systems running on two separate System/370 machines. The first AIX/370 (System A) can communicate with System B by Virtual Channel to Channel. System A can also communicate with System D by way of the LAN. System A, however, can communicate with System C by either the LAN (one relay point) or by going through System B. The choice may depend upon distance or availability of hardware.

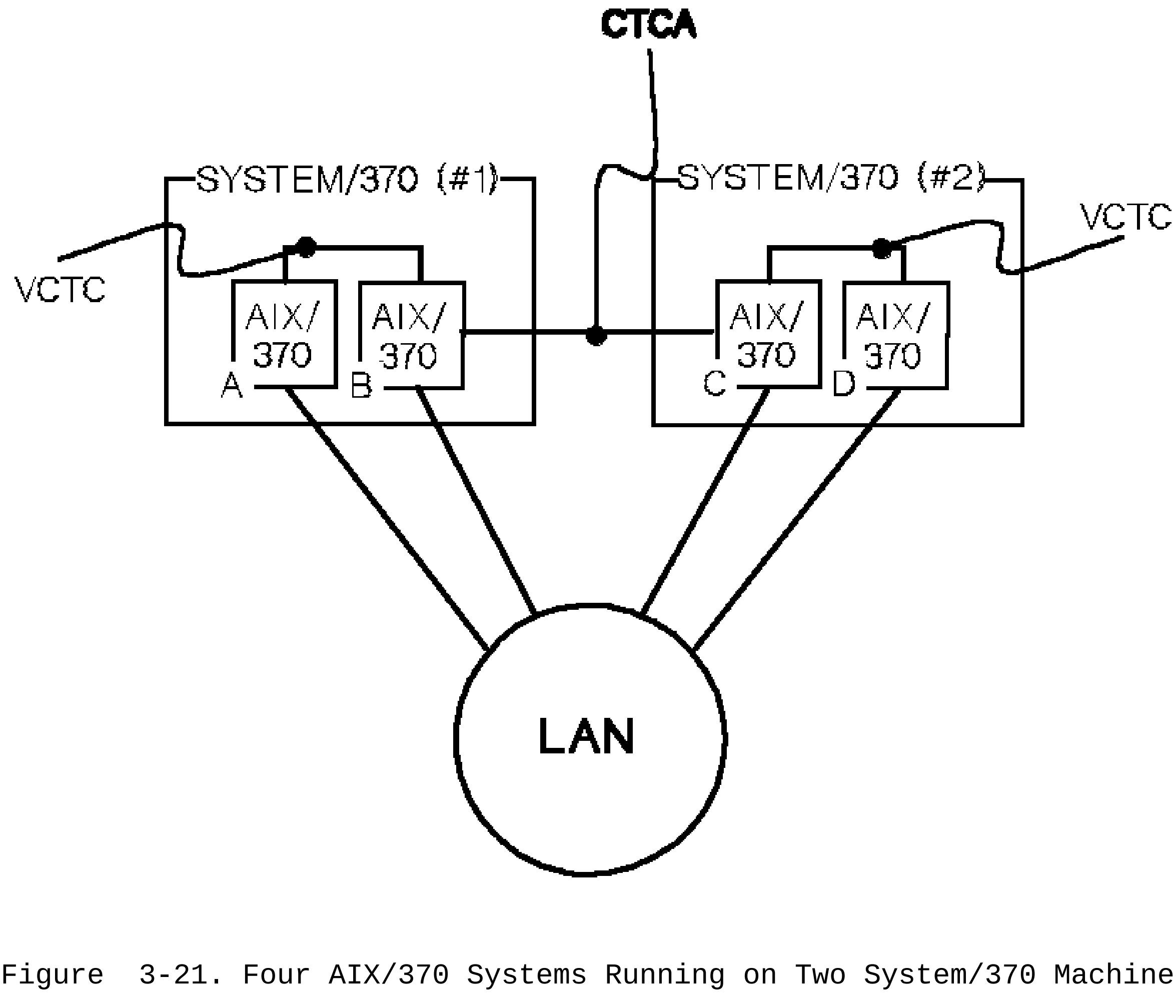


Figure 3-21. Four AIX/370 Systems Running on Two System/370 Machines

AIX users are free to log on to machines other than the one to which their workstations are directly connected by serial line or LAN channel. This ability is useful when it becomes necessary to access some special feature of that other machine, or because their normal host is too heavily loaded to run a particular job.

Nevertheless, it is standard practice to set up the home directory of a user on the machine to which there is a direct connection and to encourage the user to log on to that machine. This reduces the amount of overhead for the system in maintaining and accessing user files.
If the user's ordinary host spends considerable time off the network or is often backed up with work, it may be more efficient to replicate the files of the user on another site so that they may be processed there during such disruptions.

If the user has a choice of a direct RS-232 connection to one host, or a TCP/IP connection to others, it should be encouraged to use the RS-232 line. This results in the best response times for the user and the least overhead for the system. The TCP/IP connection(s) can be used as an alternative for those times when the regular host is down.

3.6.2 LAN Considerations

Improperly configured LAN interconnections can have a serious impact on the performance of the cluster. In addition, certain LAN devices perform better than others in terms of response time and throughput. In particular, the 8232 devices perform better than the integrated adapters of the 9370. Most LAN devices have slower throughput than the LAN that they deal with. As a result, the attachment device can become a bottleneck prior to the LAN. By providing more than one device and splitting the load over those devices, system throughput can be increased.

If a topology for the cluster is selected where packets must be relayed (forwarded) between a pair of cluster sites, then those two sites should exchange data infrequently since the relay will impact user response time.

3.6.3 Access for Users

The following list outlines the interdependencies between planning for file systems and planning for accessing files.

- Access of individual users to the cluster's resource
- Access of groups of users to the cluster's resource
- Access to sites in the cluste
- Where to have users login (distribute the users over the sites in th cluster)
- Which commands may be used by whom

Therefore, consider first investigating the potential user community, and then decide on the number of file systems and allocation of users to login cluster sites.

Data security can be facilitated by grouping the users according to their user tasks, for example:

- Program developmen
- Tester
- Information developmen
- Administratio
- End users

Any plans and decisions that have been made must be passed on to the system administrator so they can be implemented through the following functions:

Command	**Function**
adduser	defines access to the AIX/370 and defines groups
chown	restricts access rights to files and directories
chgrp	restricts access rights to files and directories
chmod	restricts the execution mode, or the access mode

Subtopics

3.6.3.1 Site Permissions

Site Permissions restrict process access to any site(s) in the cluster.

When TCF is executing in the cluster, all users have potential access to any files mounted on the replicated root file system. TCF also allows users to execute processes on any host they choose and to transfer their login location to any processor they wish. There are times when both of these freedoms need to be restricted.

Access to file systems is restricted by setting appropriate permissions on the files themselves. Access to processors is restricted by setting Site Permissions.

A site or sites can be restricted by the general public, reserving them for the exclusive use of any group the System Administrator chooses. The **adduser** command is used to set the ***Site Group*** parameter for each user added to the system. The site group used must be the name of a group in **/etc/sitegroup**, a file which gives separate names to different combinations of sites within a cluster. When this parameter has been set, the user will be restricted to the given group of sites and the system will refuse attempts by the user to login to other sites or to execute processes there. For more information on site permissions, refer to *Managing the AIX Operating System*. To inspect the site permission settings now in effect, enter:

```
pg /etc/passwd
```

The site permission settings are to be found in the **site_exec_perm** subfield of the full name field in the password file.

3.6.4 Queuing System

With the AIX/370 queuing system, controlling access to system resources is accomplished through queuing logic. System resources are identified, queues are established representing use of the resource, and algorithms are performed to select which use has precedence. An example of this might be several spool files that are to be printed; the resource is the printer and the queue is a list of spool files. An algorithm would be then used to determine which file in the list is printed first. For information on configuring the queuing system, refer to *Managing the AIX Operating System*.

In planning the queuing system, decide on the following:

Is accounting data needed for queueing activities

Which queues must be defined and which devices are to serve th queues? It will also be necessary to select the default printer queue for each cluster site.

How to tune these devices for optimum performance

Priorities of serving requests, for example, first in/first out, o should priorities be set according to job size?

How backends are defined, for example

- burst pages
- own backend process
- friendly/unfriendly backends?

How the printer classes and printer routing (System/370 only) are t be configured?

3.7 Drawing the Cluster Diagrams

Appendix B, "Examples and Completed Sample Worksheets" contains a series of four diagrams. You should use these as models for your own sketches.

In drawing your own diagrams, four steps are involved:

1. Draw the logical arrangement of the network devices and processors which will comprise the TCF cluster.

2. Fill in the software packages which will run on each site.

3. Draw another diagram which will show the same logical arrangement. Specify the name and Internet address for each site. For information on Internet addressing, refer to the TCP/IP addressing section of the *AIX Operating System TCP/IP User's Guide*.

   System/370 processors should show the virtual machines each is to run and the software running in each virtual machine. LANs are labeled as to type.

4. Draw a final diagram which combines all the features listed above. This will provide the actual entries used during the installation process.

The "Planner's Checklist" in topic A.1 is to be used after these diagrams have been completed.

4.0 Chapter 4. AIX File Systems and Minidisks

Subtopics

4.1 Contents

4.2 About This Chapter

TCF functional ability is dependent upon the nature of the AIX file system itself. This chapter explains how the replicated file system works and how to create and allocate the minidisks needed to store it. This chapter includes advice on setting up a dump facility and on backing up the file systems.

4.3 AIX File Systems and Minidisks Overview

A ***file system*** is defined as a single contiguous span of disk space on which is carried a hierarchical structure of files.

When a formatted diskette is inserted in the disk drive of an AIX PS/2 system and a suitable branching tree structure of files and directories is created on it, it becomes a file system. Such a file system is created on its own AIX ***minidisk***, and can then be appended to the existing root structure of the PS/2 file system by using the **mount** command.

The procedure for AIX/370 is similar; the prime difference is that the AIX minidisk for AIX/370 is created on a VM minidisk, rather than on the diskette used by the PS/2.

Part of the AIX installation procedure is to place different portions of the overall, global AIX file system on these separate AIX minidisks. All of these separate segments of direct access storage device (DASD) space are then joined into the ***global file system***.

Subtopics

4.3.1 Comparing VM minidisks and AIX minidisks

An AIX minidisk is a defined portion of a VM minidisk. The VM minidisk is created under VM and dedicated to the use of AIX/370. It may then be further subdivided into AIX minidisks through the use of the AIX **minidisks** command. An AIX minidisk may be any part of or an entire VM minidisk.

An AIX PS/2 minidisk involves the same concept except that it is not derived from a VM minidisk. The standard PS/2 procedure is to section the fixed disk into ***partitions***; the AIX **minidisks** command is used to further subdivide these partitions. An AIX PS/2 minidisk may be all or any portion of the fixed disk or diskette.

The AIX system requires that different parts of the overall file system be placed on separate AIX minidisks. An AIX minidisk consists of 1024-byte segments called ***blocks***. Such minidisks often take up less than the entire VM minidisk upon which they are created. It is common to maintain flexibility by leaving some space for future expansion. For more information on file systems and mounts, refer to *Managing the AIX Operating System*. To determine the DASD devices supported for AIX minidisks devoted to file systems, refer to Table 5-4 in topic 5.3.3.

4.3.2 Comparing AIX File Systems to Other UNIX File Systems

The AIX file systems have the same logical structure as UNIX System V and 4.3BSD. However, the physical image is not binary compatible.

The file system has been enhanced to make it more reliable. A **commit** mechanism has been implemented to ensure that file updates never result in a mixture of old information and new information in the file. Open files can be marked so that a **commit** can be performed at regular intervals.

The TCF cluster file system has been enhanced to support replication. Replicated file systems increase the availability of the files and directories because the file systems have up-to-date copies on several sites. Thus, if one site becomes unavailable, another site with a copy of that replicated file can provide access to that file. Replicated file systems also provide a performance advantage. Since it requires less resource to access a file locally (when a replicated copy of a file is on the local machine), it can be accessed using fewer resources than if it needed to be accessed remotely.

Compared to the UNIX System V file system, the addressing scheme within a file remains unchanged. With AIX/370, an ***inode*** contains 13 addresses. The first ten addresses point directly to data blocks. The eleventh address points to a block that contains addresses of data blocks (single indirection, that is, one level of indirect addressing). The twelfth and thirteenth addresses provide for double (and theoretically triple) indirection. AIX does not actually use this thirteenth address; it exists for future expansion. Triple indirection is not needed for the address space that the AIX/370 Operating System currently uses.

Direct access storage devices (DASDs) for AIX/370 file systems are CMS formatted into 4K (4096-byte) blocks. This structure and the addressing within a file are, however, transparent to the user, who sees a file as a sequence of bytes. The maximum size of a file is 2(31) (2,147,483,647) bytes.

For more information on file systems, refer to the *AIX/370 Administration Guide*.

4.3.3 AIX File Systems

An AIX file system is a complete directory structure, including a root directory and any sub-directories and files within it. The sizes of file systems are usually determined by the types of disks available and the particular applications for which the file systems are to be used. A file system is limited to a single logical disk. AIX divides physical disks into logical disks; each one can contain an AIX file system. The structure of a file system is completely device independent. By mounting file systems, data can be spread across several physical disks. For more information on mounting file systems, refer to the **mount** command in the *AIX Operating System Commands Reference*. For more information about file systems, refer to the *AIX/370 Administration Guide* and *Managing the AIX Operating System*.

4.3.4 AIX Minidisks Defined by the System

During installation and configuration of a typical system, the installation process defines several AIX minidisks for its basic software components and requirements. These system-defined AIX minidisks include:

/(root)
/aix370
/aix370/tmp
/u
dump
page.

For more information on the contents of these system-defined AIX minidisks, refer to "The Base System" in topic 6.3.2.1.

4.3.5 AIX Minidisks Defined by the User

User AIX minidisks can be defined in addition to the default system-defined AIX minidisks. These user-defined AIX minidisks can be used to store files, programs, or for any purpose the user may desire.

4.3.6 The Replicated Root File System

When any cluster site is installed, the AIX/370 Operating System automatically copies the root directory and its immediate branches. This entire structure, consisting of many directories and their dependent files, is replicated on every cluster site. These directories hold files which are critical to system operation, such as the file which holds the AIX kernel itself. One example of such a directory is **/bin**, which holds most of the binary files for system utilities. Each cluster site has one copy of the ***replicated root file system***. When multiple AIX sites are connected into a cluster, the operating system replicates this root file system on every site. Since these are the files from which all system functions are derived, all sites end up functioning in an identical manner. This enables the cluster to emulate a single computer. File system replication is the basis of the TCF function.

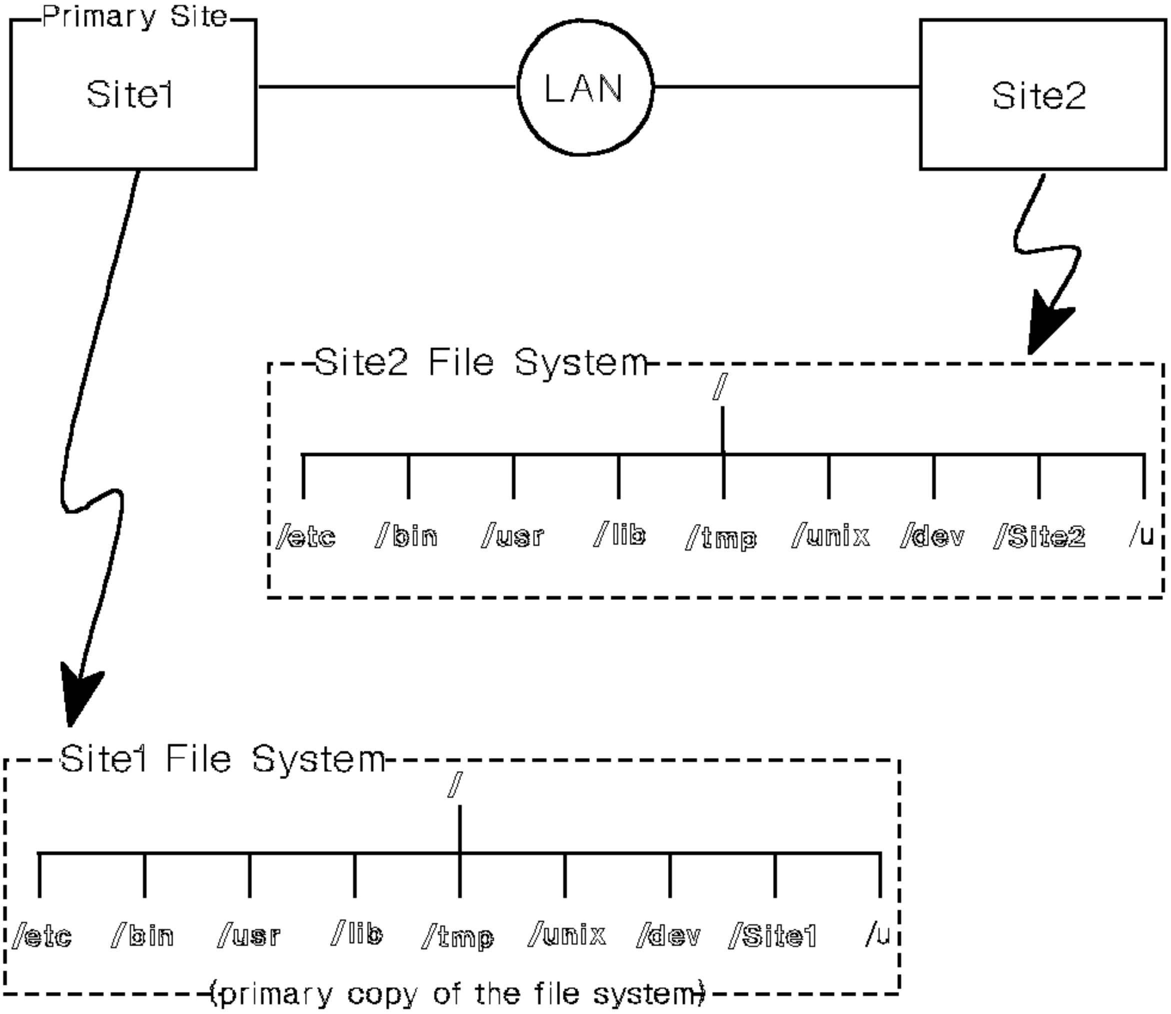


Figure 4-1. Diagram Showing Several Cluster Hosts with Same Root File System

Figure 4-1 shows two AIX sites, connected by a LAN line, both having received identical copies of the root file system. Synchronization of a replicated file system is automatic. This means that the operating system assumes the task of seeing that all copies of replicated files on all sites are kept up-to-date and identical.

There is one ***primary file system*** (stored on the primary site of the cluster) from which all other copies are made. Only the primary can be modified. All the ***secondary copies*** are read-only.

Primary copy
Each replicated root file system has a copy designated as the primary copy, which is the only copy that may be modified. It resides on the primary site and its purpose is to guarantee that file updates are kept consistent.

Secondary copy
A read-only copy of the primary copy of a replicated file system. Files in the secondary copy are automatically modified or deleted when the corresponding file in the primary copy is modified or deleted.

4.3.7 Advantages of File System Replication

Replication allows cluster sites to run independently while presenting a uniform view to all users. The AIX/370 Operating System uses replication as an integral and indispensable part of TCF operation. Replication allows all the sites in a cluster to act together as a single unit. It also provides a duplicate environment on every site so that each site has the files that it needs to execute independently, if necessary. If the site where certain files are normally stored becomes unavailable, the user can often continue to work by accessing copies of those files stored elsewhere. Replication provides an environment that is transparent to the user, whether the login site is operating as a standalone computer, or as one of the cooperating hosts in the TCF cluster.

Each cluster site contains the system files it needs to run alone if necessary, even if all other sites should leave the network. Some of these essential system files are contained in the replicated root system, and others are found in the local file system. Every cluster site has a copy of the replicated root file system, its own *<LOCAL>* file system, and several other components.

These pieces are linked smoothly together into a single continuous array. Thus all cluster sites have the ability to either run cooperatively or in a standalone environment. This replication feature allows every cluster site to be independently online or offline, without destroying the operation of the cluster.

When a host machine is installed and made part of the TCF cluster, the AIX replicated root file system is copied onto that host from the primary site of the cluster. Then various other file systems are appended to it by the process called mounting. Each of these mounted file systems is contained on a separate AIX minidisk. On a large AIX/370 system, each AIX minidisk may be as large as an entire VM minidisk. On an AIX PS/2 host, each file system will occupy some smaller area, a partition on the fixed disk or some portion of a diskette in a certain drive.

Figure 4-2 is a representation of the logical structure of the root file system with additional AIX minidisks mounted on it. User file systems are generally mounted at the point called /u, or at points immediately dependent upon it.

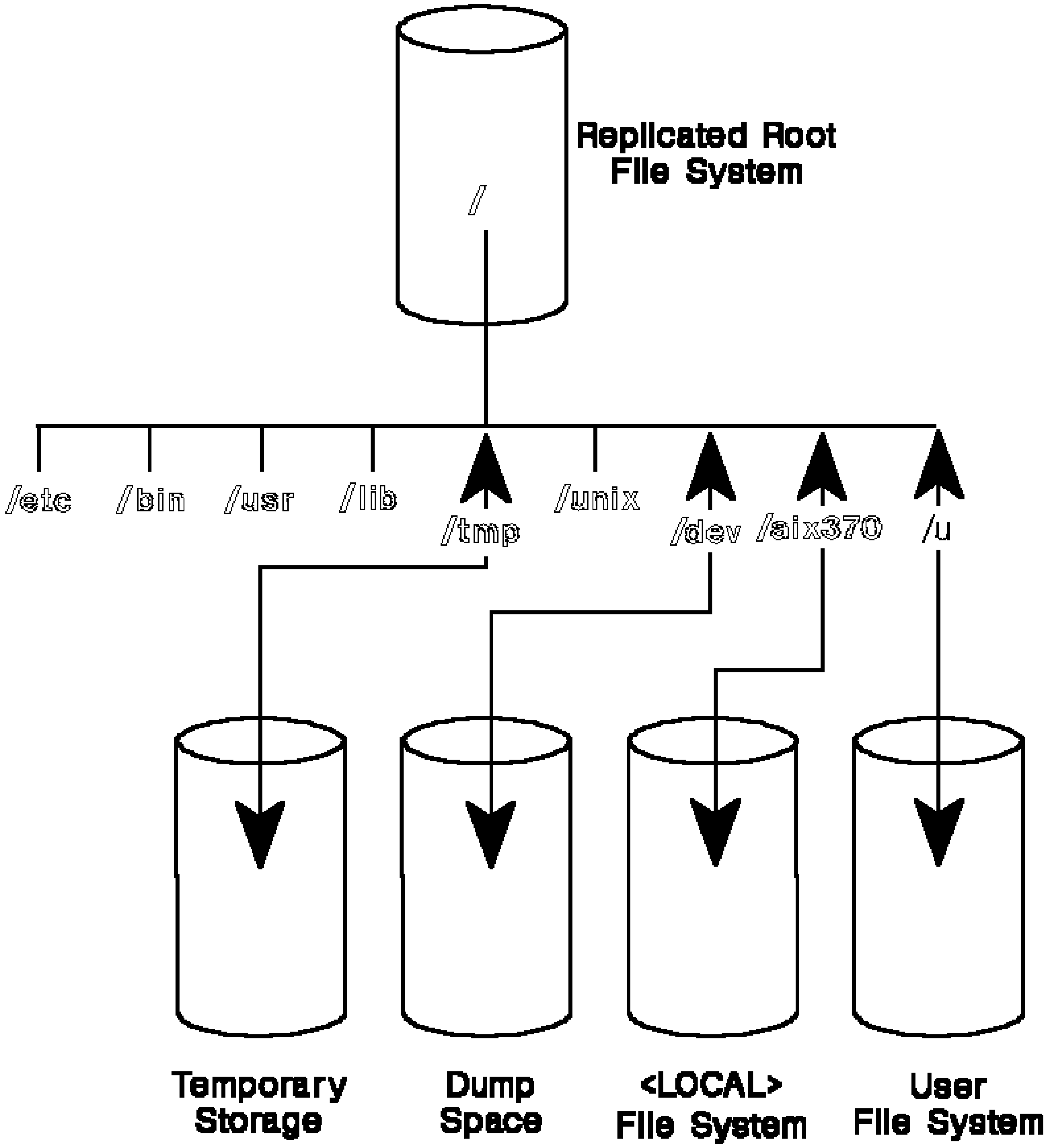


Figure 4-2. Logical Structure of the Root File System

A temporary storage file system is created on a separate AIX minidisk which is then mounted on the empty stub called **/tmp**.

The point called **/dev** is used to mount device drivers (programs that handle input/output operations to the system's various peripheral devices). One notable component found in this group is **/dev/dump**, which directs a storage dump to the dump AIX minidisk in the event of a system halt. The **/dev/dump** component usually sends its output to the dump partition of the fixed disk on a AIX PS/2 machine and to an AIX minidisk on a AIX/370 machine. The local file system (*<LOCAL>*) is appended to the root at the point called **/aix370**.

These components, taken together, constitute what is called the global file system. This includes all the files known to the AIX operating system on that host. These file systems are integrated smoothly together so that they appear to be one continuous directory structure.

4.3.8 The <LOCAL> File System

The *<LOCAL>* file system is named for the particular cluster site whose files it holds. It includes site-specific system information, such as accounting and administrative files that track system usage, and printer spool files for that site. The *<LOCAL>* file system may also contain the binary files for the application programs which are specific to that site. One strategy of system administration is to assign all the members of a particular group to the same host as their regular logon site. In this case, the application programs needed by those users may be stored on that host, and the data files, with which they usually work, may be stored there also. There may be copies of all these files elsewhere on the network so they can work when their usual host is down. The local file system is appended to it at the point called */sitename* (often referred to as */machinename* or */<LOCAL>*). Both of these are shorthand ways of naming the point where the local file system is mounted.

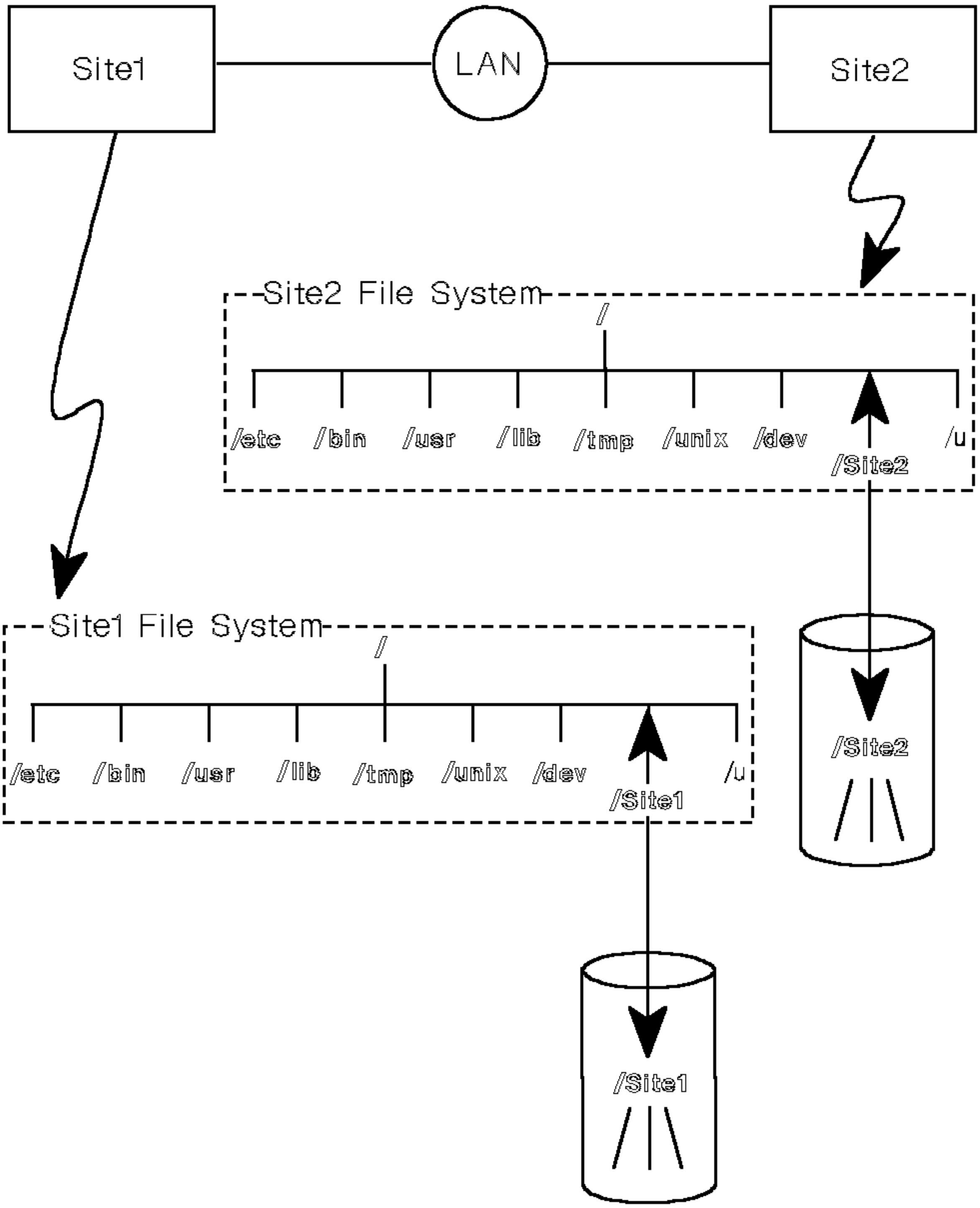


Figure 4-3. Diagram Showing Two Hosts with Different <LOCAL> File Systems Mounted

Figure 4-3 shows two AIX sites sharing an identical root file system, but each having its own local file system mounted.

When the AIX/370 operating system is installed, this mount point is called **/aix370**. During installation, this directory is given the actual name of the machine on which it is being installed. The file system which is to be mounted there will be given exactly the same name. The two are then joined by the **mount** operation.

4.3.9 User File Systems

In a TCF cluster, the system administrator usually sets up the file systems that store users' home directories on a number of different sites. It is common to spread these file systems evenly over the available processors rather than gather them all on a single site. These user file systems are usually mounted on the replicated root system at mount points with names similar to: */u1*, */u2*, */u3*, and so on.

When these computers are running in a cluster environment, the user file systems mounted for each host are mounted on empty directories given unique names. TCF makes all of these user file systems available from any point in the cluster, with access to the actual files controlled by file permissions.

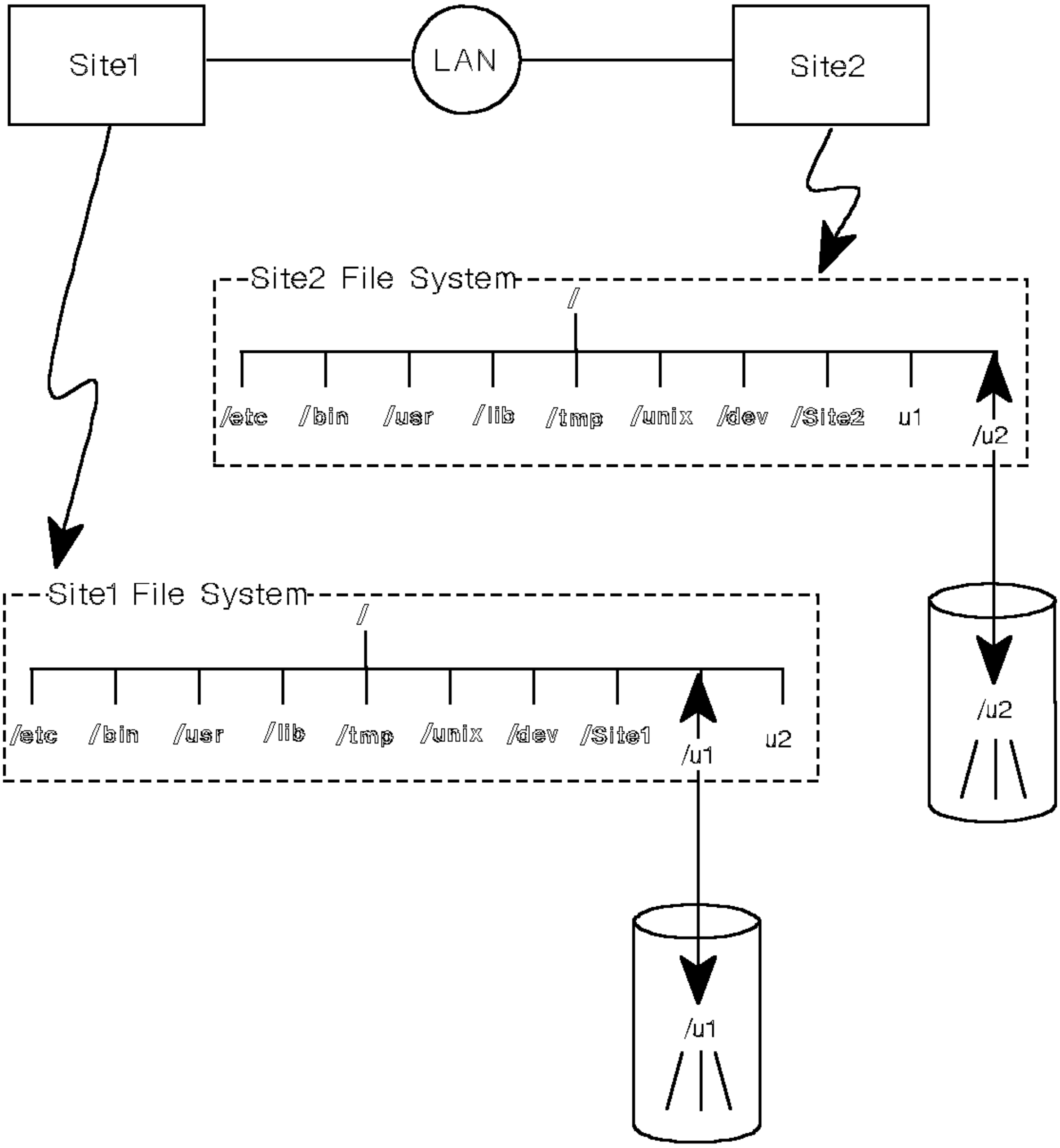


Figure 4-4. Diagram Showing Two Hosts with Two Sets of User File Systems Mounted

Figure 4-4 shows two AIX sites sharing an identical root file system, but each having its own user file system mounted.

4.3.10 Using Hidden Directories to Organize Binary Files

The ***hidden directories*** mechanism leads to ***command name transparency***, one of the main user benefits of the AIX operating systems. The user can issue a single command and get the desired function to execute, no matter what type of processor happens to be doing the job. Users do not need to consider, or even know, what type of processor they were logged onto.

The AIX **ls** command lists the contents of a directory. When the operating system executes this command, it actually runs a 370-style program on an AIX/370 machine, and a 386 module on an AIX PS/2 machine. Hidden directories are essential to this behavior.

A TCF cluster is usually composed of both AIX/370 and AIX PS/2 machines. The primary copy of the replicated root system will store program modules for both processor types. AIX provides a mechanism called hidden directories to support this heterogeneity (mixture of different program modules). The **-H** option to the **ls** command allows you to see the entries in a hidden directory as separate items. For instance, if a long listing of the hidden directory **/bin/awk** is requested, the results will look like this:

```
ls -lH /bin/awk
total 240
-r-xr-xr-x  2  bin  bin   123880   Mar 2 14:14   i370
-r-xr-xr-x  2  bin  bin   106864   Mar 2 14:14   i386
```

This shows that the **/bin/awk** program is actually two separate modules.

These files contain binary data, the program files for applications and system utilities. Binary data is machine dependent. A program compiled on a 386 processor will not execute on a System/370 machine. Thus it is important to know which machine generated each binary file in order to run the right program on the right processor.

Hidden directories are often used to organize binary data so that the proper program is used on each respective processor.

4.3.11 Primary Versus Secondary Copies of the Replicated Root

A hidden directory is part of the replicated root file system. This entire file system is replicated on each cluster site, but not every entry in every hidden directory will be copied to every site in the cluster. Each hidden directory contains program modules for different types of processors and the system will usually copy to each site only those modules needed by that processor.

The primary copy of the replicated root system must store every program file necessary for system operation anywhere in the cluster. In this way, the 386 version of a certain program can be updated once, and it will automatically be propagated to all other 386 sites. Likewise, 370 modules must be stored in the primary copy so that, from this one centralized location, corrected modules can be propagated to all AIX/370 sites.

The secondary copies of the root file system do not need to contain program modules for any other type of processor than the machine to which they are copied. For this reason, the replicated root file system of the cluster's primary site will be larger than most of the secondary copies replicated from it. The primary site of the cluster is usually the first AIX/370 machine that is installed. Thus it is likely that succeeding installations of AIX/370 may require less disk storage space for the base operating system, since they usually store only 370 modules. This fact will become important later, when planning the amount of VM minidisk space to be allocated to each system in the TCF cluster.

4.3.12 The AIX /dev/dump Facility

Device drivers are special AIX files that may be treated as if they were ordinary files. Input may be sent to them or received from them. What they actually do, however, is control peripheral devices.

For example, **/dev/dump** controls the flow of information sent to disk or tape when a system halt occurs. The normal device choices are the dump partition of the fixed disk for AIX/ PS2 and an AIX minidisk for AIX/370.

Before installation, decide how much dump space is to be allocated. The minimum is enough AIX minidisk space for one dump; this would be the size of the virtual machine. It is often wise to provide space for several dumps so that the contents of memory may be compared under different experimental conditions.

The area reserved for **/dev/dump** is allocated by the **maint** facility in the same way that any other AIX minidisk is allocated. For more information on the **/dev/dump** area, refer to **dump** entry in "The Base System" in topic 6.3.2.1.

4.3.13 The AIX /dev/swap Facility

AIX maintains its own paging methods in addition to paging provided by VM. The **/dev/swap** program is a device driver which provides access to an AIX minidisk mounted at this point. This is accomplished through a symbolic link (refer to "Symbolic Links" in topic 4.3.14).

A paging area must be specified even though AIX/370 runs under the control of VM. One paging area can be allocated on a DASD or on minidisk that is defined as paging volume, or one or more paging area(s) can be allocated on other volumes that also contain file system space. For each system, allocate at least 4096 pages of paging space. It is recommended to add at a minimum of ten pages for each user expected to work with AIX/370.

4.3.14 Symbolic Links

A ***symbolic link*** is a substitution mechanism. A file or directory is known by one name, and treated by the system as a logical unit positioned at a certain spot in the global file system. The operating system uses another component as a substitute. This other component is actually located at some different spot in the file system and called some other name.

AIX makes extensive use of this mechanism. One significant use is the substitution of <*LOCAL*> components for standard parts of the file system.

For instance, the file **/unix** contains the kernel of the operating system. The kernel is a hardware-specific item. It must be different for each machine on which it runs. Therefore it is convenient to have the specific kernel needed by each machine located on the local AIX minidisk. In this way, an AIX/370 machine will run a 370 kernel, an AIX PS/2 machine will run a 386 kernel, and both will appear to be found at the same point on the file system. Therefore, the **/unix** file is a symbolic link to the <*LOCAL*>**/unix** file.

Likewise, the **/dev/dump** file and **/dev/swap** file are symbolic links to the <*LOCAL*>**/dev/dump** file and <*LOCAL*>**/dev/swap** file. Each of these files is a device driver which provides access to further AIX minidisk space. The **/dev/dump** file gives the system access to a separate AIX minidisk to be used as dump space. The **/dev/swap** file provides access to an AIX minidisk used for paging space.

Compare Figure 4-5 to Figure 4-2 in topic 4.3.7. Figure 4-2 shows the logical arrangement of the global file system. Temporary storage, dump space, and paging space are treated as if they were all separate minidisks attached to the root system directly at the points. The **/unix** file appears to be directly mounted on the root.
Figure 4-5 shows the actual arrangement of the global file system. Temporary storage is actually situated in its own minidisk on the local disk. The functional kernel is the <*LOCAL*>**/unix** file. The device drivers for dump space and paging space are actually files on the <*LOCAL*> disk, giving access to separate minidisks for those purposes. The Installation/Maintenance (I/M) system is separate, on its own VM minidisk. User file systems may be situated on physically separate disks, as shown, or they may be also found on the <*LOCAL*> minidisk.

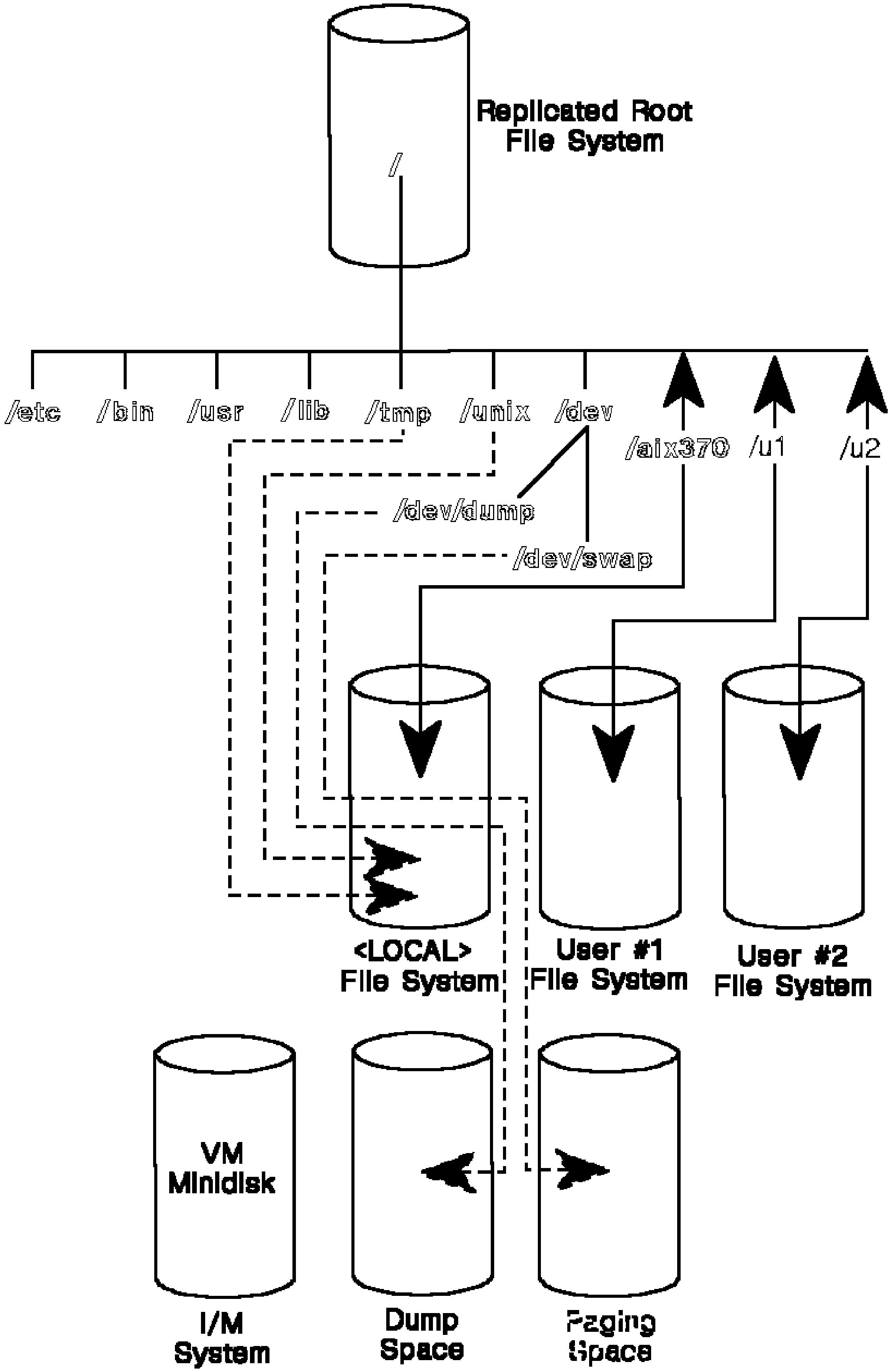


Figure 4-5. Actual Structure of the Root File System

4.4 File System Backup

File systems are an important resource. Backing up the file systems ensures that files are not damaged or lost.

Decisions must be made about:

- Which files or file systems need to be backed up
- How often to have backup copies produced
- Which backup utility program and backup scenario are best suited fo the installation environment and backup goals?
- Where the backups will be kept and how often the tapes will be reused

Consider backing up:

- System file systems (AIX/370 and AIX PS/2
- User file systems (user files
- Maintenance syste
- Critical VM data

For more information on backing up file systems, refer to *Managing the AIX Operating System*.

Subtopics

4.4.1 AIX/370 System Backup

To produce backups of data, AIX/370 includes the following utilities:

Command	**Function**
backup/restore	Copies entire file systems or individual files and is also useful for incremental backups.
tar	Copies files, individually or in groups.
ff/find/cpio	Copies file systems and performs incremental backups.
DDR	Backs up binary images of VM minidisks (a VM tool).

With these utilities, backups can be made from disk to disk (on spare disks) and from disk to tape. For more information on the VM tools (for example, **DDR**), refer to the *VM/SP Planning Guide and Reference* and *VM/XA System Product Planning and Administration*.

4.5 File System Planning

In planning for the file systems that are to be used at your installation, consider the following (depending on requirements, this list may not be all-inclusive):

Before AIX/370 or AIX PS/2 can be used in multi-user mode (the mod required for end-users), decide:

- Number of file systems needed?
- Size and number of files in each file system?
- Number of file systems which will change, and which will therefore need to be backed up regularly?
- How to distribute the file systems across the cluster?
- How to distribute the file systems across the available disk space?
- How file systems should be accessed by individual users or by groups of users?
- How file systems should be protected against unauthorized access?

After defining the user file systems, decide

- Which procedures must be followed for file system backup (for example, daily, or weekly incremental backup)?
- Which procedures must be followed to maintain sufficient free disk space?
- Which file systems must always be mounted (such file systems must be defined in the **/etc/filesystems** file)?
- What file system check procedures are to be used (the file systems must be defined in the **/etc/filesystems** file)?

  It is normal to run the **fsck** command on any file system which might have changed, before that file is mounted. Any file system which was last mounted read/write should be checked. For more information on file systems, refer to the *AIX/370 Administration Guide* and *Managing the AIX Operating System*. For more information on the **fsck** command, refer to the *AIX Operating System Commands Reference*.

5.0 Chapter 5. Requirements for AIX/370

Subtopics

5.1 Contents

5.2 About This Chapter

The AIX/370 Operating System runs as a guest under VM. In order to plan for the installation and day-to-day running of AIX/370, the planner must be completely aware of the needs of the program. This chapter discusses the following AIX/370 requirements:

- Softwar
- Hardwar
- Devices supported by AIX/370

5.3 AIX/370 Requirements

The following information outlines the requirements that must be met in order to execute AIX/370 as a guest under VM. These requirements are comprised of:

Softwar
Hardwar
Supported devices

Subtopics

5.3.1 Software Requirements for AIX/370

AIX/370 can coexist with any other VM guest systems and CMS. AIX/370 operates as a guest system under one of the following required IBM programs:

IBM VM/SP, Program Number 5664-167, Release 5.0 or late (1), on an IBM System/370 processor.
IBM VM HPO Program Number 5664-173, Release 5.0 or later¦, on an IB System/370 processor.
IBM VM/XA SP Program Number 5664-308, Release 1.0 or late (2), on an IBM System/370XA processor.

Notes:

1. AIX/370 does not support Preferred Machine Assist (PMA) or single processor mode.

2. For Japanese locales, AIX/370 Version 1.2.1 requires VM/XA SP.

For normal service support, the following must be available:

Environmental Recording Edit and Print program (EREP), Program Numbe 5749-010 (feature #5022 or #5023)
Device Support Facilities (DSF), Program Number 5747-DS1

(1) Release 5 must be at service level 0513; Release 6 must be at service level 0601.

(2) Release 1 must be at service level 0102; Release 2 must be at service level 0205 with fix VM39289 applied (VM39289 corrects wedge problems).

5.3.2 Hardware Requirements

The minimum hardware configuration for the primary cluster site AIX/370 guest system is outlined in Table 5-1. These resources are used by the AIX/370 guest system exclusively, except where noted.

The hardware requirements for VM to properly execute AIX/370 as a guest system varies with the version of VM used to support AIX/370 and the other virtual machines that are to be supported. There are VM hardware resources that may be shared with other guests besides AIX/370. Also determine which VM support packages are being used and what the machine requirements are in order to determine the total VM machine requirements.

Table 5-1. Hardware Requirements for the Primary Cluster Site AIX/370 Guest System

Resource	**Hardware Requirements**
Processor	AIX/370 supports processors both in System/370 mode and System/370 Extended Architecture (XA) mode. AIX/370 will execute on any IBM processor which supports the required level of VM/SP, VM/HPO, or VM/XA. For more information on the required levels of VM, refer to "Software Requirements for AIX/370" in topic 5.3.1. **Note:** For Japanese locales, AIX/370 Version 1.2.1 requires a System/370 XA processor with VM/XA SP.
Processor Storage	Proper operation of AIX/370 depends on the physical machine having enough real storage and extended storage to avoid excessive VM paging. Total the sizes of the AIX/370 guest system virtual machines on a single VM system. The total sizes of the AIX guests on a single VM system should only be less than the real memory if it is a v=r system. This does not apply to v=v systems. Configure the AIX/370 guest system virtual machine with a minimum of 6Mb and a maximum of 16Mb for VM/SP of storage.
DASD	The DASD types supported for the Base System and Installation/Maintenance System are: ***FBA*** — 3370, 9332, 9335 ***CKD*** — 3380, 3390 DASD space may be allocated on an entire disk or a VM minidisk. To determine the capacities of the DASD devices, refer to *VM/XA System Product Planning and Administration* and the *VM/SP Planning Guide and Reference*.
DASD Space	The initial AIX/370 cluster site requires a minimum of 325 cylinders of 3380 DASD space or 390,000 FB-512 blocks of DASD space. This does not include space for

	non-system data storage.
Tape	One tape device which may be any of the following types: 3420 3480 3422 8809 3430 9347
Console	One VM operator console (usually a 3270 display terminal). One AIX/370 guest system console (3270). This station is used as AIX/370 guest system console; it is for installing and operating AIX/370 as a guest system. **Notes:** 1. Optionally, VM and the AIX/370 guest system can share the same 3270 as the operator console by executing AIX/370 under the OPERATOR userid, but this is not recommended unless there is a site terminal shortage. 2. A PS/55 running PS/55 as 3270 software may be used as a console on AIX/370 Version 1.2.1 systems in Japanese locales.
Printer	AIX/370 uses only VM spooled printers. **Note:** Without the appropriate print chain, IBM printers will not print nor translate (to other printable characters) the \ (backslash), ` (backquote), { and } (braces), [and] (brackets), ~ (tilde), and ^ (circumflex) characters.

For each secondary cluster site AIX/370 guest system on the same processor, the hardware requirements outlined in Table 5-2 must be met.

Table 5-2. Hardware Requirements for Each Additional Secondary Site AIX/370 Guest System on the Same Processor

Resource	Hardware Requirements
Processor Storage	Proper operation of AIX/370 depends on the physical machine having enough real storage and extended storage to avoid excessive VM paging. Total the sizes of the AIX/370 guest system virtual machines on a single VM system. The total sizes of the AIX guests on a single VM system should only be less than the real memory if it is a v=r system. This does not apply to v=v systems. Configure the AIX/370 guest system virtual machine with a

	minimum of 6Mb and a maximum of 16Mb for VM/SP of storage.
DASD Space	Subsequent AIX/370 cluster sites on the same processor may require: 500 cylinders of 3380 DASD or 600,000 FB-512 blocks of DASD space for a secondary copy of the root file system. 500 cylinders of 3380 DASD or 600,000 FB-512 blocks of DASD space for a backbone copy of the root file system. **Note:** This does not include space for non-system data storage and assumes that the I/M System will be shared with the other AIX/370 guest system.

For each secondary cluster site AIX/370 guest system on a different physical processor, the hardware requirements outlined in Table 5-1 must be met with the exception of DASD space which is outlined in Table 5-3.

Table 5-3. DASD Requirements for a Secondary Cluster Site AIX/370 Guest System on a Different Physical Processor

Resource	**Hardware Requirements**
DASD Space	Subsequent AIX/370 cluster sites a different physical processor may require: 585 cylinders of 3380 DASD or 702,400 FB-512 blocks of DASD space for a secondary copy of the file system. 585 cylinders of 3380 DASD or 702,400 FB-512 blocks of DASD space for a backbone copy of the file system.

5.3.3 Devices Supported by AIX/370

Table 5-4 lists the all IBM devices that are supported by AIX/370. Default device addresses defined by the AIX kernel are listed in Appendix C, "AIX/370 Default Device Addresses."

Table 5-4. Devices Supported by AIX/370

Device	**Model**
DASD	***FBA***: 3370, 9332, 9335 ***CKD***: 3350, 3375, 3380, 3390 DASD space does not need to be allocated on a single device. Several different DASDs, even of different types, may be used to store data. To determine the capacities of the supported DASD devices, refer to *VM/XA System Product Planning and Administration* and *VM/SP Planning Guide and Reference*.
Tape	3420, 3422, 3430, 3480, 8809, 9347 If a tape device was not dedicated to the AIX/370 guest system, VM must attach tape devices as needed.
Printer	AIX/370 uses only VM spooled printers. **Note:** Without the appropriate print chain, IBM printers will not print nor translate (to other printable characters) the \ (backslash), ` (backquote), { and } (braces), [and] (brackets), ~ (tilde), and ^ (circumflex) characters.
Controller	All the controllers for the previously mentioned DASD devices are supported with the exception of caching controllers that are used in cache mode.
Network Devices	Depending on the needs of the installation and the processor being used, one or more of the following network devices are required to connect an AIX/370 processor to the user environment. The following LAN interfaces are supported: All models of the 8232 (requires a channel card when used on the 9370) The 9370 LAN Feature (Ethernet and Token-Ring) integrated adapters of the 9370 machines Between AIX/370 machines, TCF is supported over Virtual Channel-to-Channel (VCTCA) and over a real

	CTCA or 3088. In the case of a 3088, a single physical address of the 3088 at each end must be dedicated to the AIX/370 virtual machine.

6.0 Chapter 6. Installation Planning

Subtopics

6.1 Contents

6.2 About This Chapter

This chapter describes planning for software installation, which includes:

- Determining the amount of virtual storage for each AIX/370 gues system virtual machine
- Using the AIX minidisk space requirement worksheet
- Using the VM minidisk space requirement worksheet
- Other installation planning issues

6.3 Planning for Software Installation

Before the AIX/370 kernel or any other AIX/370 programs are installed, the AIX/370 guest system virtual storage and VM Direct Access Storage Device (DASD) requirements must be determined.

Worksheets that aid in determining DASD requirements can be found in Appendix A, "Worksheets." These worksheets are provided to make sure that there is enough DASD space (VM and AIX minidisk) to support the operating system and licensed programs that are intended to be used. Also, these worksheets are used when setting up the AIX/370 guest system virtual machine. Photocopy the worksheets prior to completing them to ensure that there will be a blank original for future uses. After these worksheets are completed, they should be passed on to personnel responsible for the set up of the AIX/370 guest system virtual machine.

Completed sample worksheets can be located in Appendix B, "Examples and Completed Sample Worksheets." Use these worksheets as an aid when filling out the worksheets for your installation.

Note that not all of the real storage in the CPU is for the AIX/370 guest. At the very least, VM must be run as well and VM will use some processor storage for its own needs. In most cases, AIX/370 will share available processor storage with VM and other guest operating systems. Each such guest operating system will use some storage for its own purposes.

Subtopics

6.3.1 Virtual Storage and Paging

AIX/370 supports virtual storage and paging (the page size is 4096 bytes.). Processes operate in either 10Mb of virtual address space for System/370 machines or 768Mb of virtual address space for System/370XA machines. A System/370XA machine can execute a System/370 mode process.

Each process has its own virtual address space. User programs occupy space in the lower portion of the process space; the kernel is located in the higher address portion of the process space.

Subtopics

6.3.1.1 Estimating Virtual Storage

The minimum configuration for AIX/370 is 6Mb of virtual storage, in addition to VM requirements. The amount of virtual storage and real storage required by the system depends on the number of system users, the number and size of licensed programs used, and the response time desired. As the number of users increases, the storage parameters may need to be increased to maintain performance.

6.3.2 Space Allocation on DASD for AIX/370

The VM DASD requirements consist of the total of the VM minidisks needed for the Base System, the Installation/Maintenance (I/M) System, and AIX minidisks.

A VM minidisk is, all, or a logical subdivision of, a DASD device, that has its own address, consecutive storage space for data, and an index or description of stored data. The VM minidisks are divided into segments which are expressed as number of FB-512 blocks for Fixed Block Architecture (FBA) type DASD devices and cylinders for Count Key Data (CKD) type DASD devices. The AIX/370 system uses VM minidisks to store the Base System, Installation/Maintenance (I/M) System and AIX minidisks.

AIX minidisks consist of 1024-byte segments called blocks. All references to AIX minidisks size are expressed as number of blocks. The physical records on the disk are actually 4096 bytes, each containing four blocks. During installation of a typical AIX/370 system, the installation process defines several AIX minidisks as part of the Base System. For more information on system defined AIX minidisks, refer to "The Base System" in topic 6.3.2.1. Besides the system-defined AIX minidisks, user AIX minidisks for data storage may also be defined. To determine the size of the VM minidisks which are used for user AIX minidisks, the size of all the user AIX minidisks must first be determined then added together for the total.

Subtopics

6.3.2.1 The Base System

AIX/370 requires at least one VM minidisk to use as the Base System volume, that is, as the device from which to load AIX/370. The Base System volume also contains the system-defined AIX minidisks. For information on supported DASD devices for the Base System, refer to Table 5-1 in topic 5.3.2.

The layout of the Base System volume is determined at installation time based on the input provided by the installer. The following AIX minidisks are defined as part of the installation process:

AIX Minidisk	Contents
/ (root)	The **/** minidisk, called the root minidisk, is the highest level file system of the AIX/370 operating system. This minidisk can also contain some of the system files and licensed program files, but should not be used for personal user files.
/aix370	**/aix370** is the default for the *sitename* minidisk. This is the local file system used to store the frequently updated system files.
/aix370/tmp	The **/aix370/tmp** minidisk is a file system used for temporary files. If large files are archived, sorted, or edited, the size of the **/aix370/tmp** minidisk must be increased to accommodate these large files.
/u	The */u* minidisk is created for storage of user files. The amount of AIX minidisk space to allow for */u* depends on how many users the system will be supporting and the licensed programs that are planned to be used.
dump	The dump minidisk contains operating system storage dumps. This information is helpful in case problems are experienced or diagnostic service is required at the site. The dump size depends on the amount of virtual storage that has been allocated to the AIX/370 guest system virtual machine. The **/dev/dump** file in **/aix370** is used to access the dump AIX minidisk physical location. When installed, dump minidisk must provide sufficient space for at least one storage dump. If more space is needed for storage dumps, allocate more dump minidisk space by using the **maint** facility. For more information on allocating additional dump storage space, refer to the *Managing the AIX Operating System*. As initially configured, the dump minidisk can be defined on another disk, and relinked to the special file (**/dev/dump**) for the new dump minidisk. It will also be necessary to regenerate the kernel with the new dump minidisk device indicated.
page	Paging is the act of copying virtual storage into or out of the available real storage. The default page minidisk sizes are determined by the DASD configuration and the amount of real storage. These defaults can be changed when the operating system is installed, if desired. The **/dev/swap**

file in **/aix370** is used to access the page minidisk physical location.

Although the system-defined AIX minidisks are defined as part of the Base System, they may be installed on their own VM minidisks. VM minidisk virtual addresses 342 to 345 (for FBA devices) and 392 to 395 (for CKD devices) may be defined in the AIX/370 guest system virtual machine and used in part or totally for any of the system-defined AIX minidisks.

6.3.2.2 The Installation/Maintenance (I/M) System

As part of the installation process for AIX/370, an Installation/Maintenance (I/M) System is read in from the distribution tapes. The I/M System is used to complete the installation and can then be used to service the AIX/370 system in emergencies. The I/M System can be put on the Base System volume, but putting it on its own VM minidisk is preferred. This allows the maintenance system to be more efficiently protected against corruption and backup is straightforward. The administrator should keep the I/M system up to date so it is ready to be used in emergency situations. For information on supported DASD devices for the I/M System, refer to Table 5-1 in topic 5.3.2.

6.3.2.3 Other AIX and VM minidisks

Space can be allocated on the VM minidisks that do not contain the Base System or the I/M System for the following file systems:

Minidisk	**Contents**
User AIX minidisk(s)	User AIX minidisks can be defined in addition to the default system-defined AIX minidisks. These user-defined AIX minidisks can be used to store files, programs, or for any purpose the user may desire. Some licensed programs require that a separate AIX minidisk be defined for their files. For some large programs, it is recommended that separate AIX minidisks be defined. User files can be stored on the **/**, **/aix370/tmp**, or **/aix370** minidisk. However, if the operating system is ever reinstalled using the replace option rather than the update option, these minidisks may be completely replaced, and user files lost. It is recommended that user files be placed only on AIX minidisks defined for user files and mounted to those mount points reserved for user file systems (*/u1*, */u2*, */u3* ...). **Note:** Approximately five percent of the space is used for indexes, so the actual space available for data is about 95 percent of the total minidisk space.
Service file system	This file system is used to save backup images of fixes in case it becomes necessary to remove a fix that has been installed with **updatep**. If this data is kept in a separate file system, then it only needs to be mounted when service functions are being performed. The isolation of this data protects it against accidental damage and keeps these old versions from being replicated to each site with the rest of the root file system. For more information on the service file system, refer to *Installing and Customizing the AIX/370 Operating System*.

These file systems will not actually be created at this time. The creation process is done after AIX/370 is up and running. The AIX **minidisks** command is used to:

Allocate the space for the file system on a previously defined V minidisk.

Create a mount point for the file system by modifying the AIX roo file system.

However, the space on the VM minidisks should be allocated at this time. For information on supported DASD devices for data storage, refer to Table 5-4 in topic 5.3.3.

6.3.2.4 Calculating DASD Requirements

The following worksheets are provided in Appendix A, "Worksheets" in topic A.0:

"AIX Minidisk Worksheet" in topic A.2
"VM Minidisk Worksheet" in topic A.3.

The AIX minidisk worksheet is provided so the AIX minidisk requirements (in addition to the system-defined AIX minidisks) can be calculated for a given set of programs or users. Several photocopies of this worksheet may be required depending on the number of programs to be installed and how many user AIX minidisks are required.

The VM minidisk worksheet (Table A-2 in topic A.3) helps to verify that the number and size of VM minidisks chosen will support the programs and user AIX minidisks. In addition, the worksheet is used for determining the placement of VM minidisks on DASD devices. The total storage requirement from all AIX minidisks worksheets is to be included in this worksheet in either the Base System or the AIX minidisks rows.

Note: Use photocopies of the worksheets to ensure that there are blank originals for future use.

<u>Completing the AIX Minidisk Worksheet</u>: The AIX minidisk worksheet is used to calculate the total AIX minidisk requirements for the AIX/370 operating system. These requirements include AIX minidisk space for:

User file system
Program
Additional space for system-defined minidisks

Fill out the AIX minidisk worksheet ("AIX Minidisk Worksheet" in topic A.2) by performing the following steps:

1. List the programs that are to be installed along with their block-sizes in the appropriate columns.

2. List the user file systems that are to be installed with their block-sizes in the *user* column.

   Note: If a large number of file systems are to listed or if the file systems are to be placed on their own AIX minidisk, list them on a separate worksheet.

   If the specific block size of the user file systems are not known, use the following as a guideline to determine the amount of AIX minidisk space to allocate:

   a. ***Overhead*** = (Number of files * 512) * 1.10

      This calculates the overhead necessary for file system storage (for example, space used for inodes and superblocks). A padding factor of 10% is added for safety.

   b. ***Data Storage*** = (Number of files * Average file size) * 1.10

      This calculates the amount of space devoted to actual data storage. A padding factor of 10% is added.

c. ***Total Space*** (in 1024 blocks) = (***Overhead*** + ***Data Storage***) / 1024

d. Record the Total Space amount in the AIX minidisk *user* column of the AIX minidisk worksheet.

It must be emphasized that this is a guideline and gives a starting point only. From this point on, it will be up to the System Administrator to observe carefully whether the amount of space allocated meets the actual needs, and to adjust the future use of these formulae accordingly.

The way to apply these guidelines in actual practice is to break users down into logical groups (for example, programmers, text processors, and accounting users), then inspect their files carefully to arrive at an average file-size for each group. Apply the second, third, and fourth calculations to each group; this will yield Total Space values for each group. These values will provide the basis for the creation of large file systems devoted to each department or group.

3. List any additional space that is required for the system-defined AIX minidisks in the appropriate column. The default sizes of the system-defined AIX minidisks can be found in "The Base System" in topic 6.3.2.1.

4. Sub-total the results of each AIX minidisk column (**/** (root), **/aix370**, **/aix370/tmp**, *user*, **dump**, and **page**) for each AIX minidisk worksheet.

5. If additional AIX minidisk space is required for future expansion, record it in the Future Uses row.

6. Total the results of each AIX minidisk column (**/** (root), **/aix370**, **/aix370/tmp**, *user*, **dump**, and **page**) for each AIX minidisk worksheet.

   If more than one AIX minidisk worksheet was completed, add together the column totals for all the worksheets.

<u>Completing the VM Minidisk Worksheet</u>: The VM minidisk worksheet is used to calculate the total VM minidisk requirements for the AIX/370 operating system. These requirements include:

Determining if AIX minidisks are to be placed on

- The Base System VM minidisk (default)
- Their own VM minidisk
- VM minidisks in groups (for example, the **page** and **dump** system-defined AIX minidisks may be place on one VM minidisk while the other system-defined AIX minidisks are place on the Base System minidisk).

Determining the VM minidisk siz

Recording the Volume ID or real address of where the VM minidisk is t be placed

Recording the starting block or cylinder (or noting if the entire dis is to be used).

After completing the AIX minidisk worksheet(s), follow these steps to calculate the VM minidisk requirements:

1. Determine the type of site that is being installed from Table 6-1 and enter the appropriate values for the Base System and the I/M System.

   Note: The following table assumes that you are installing PS/2 LPP's in your cluster.

   If you are only installing AIX370 use the least of these values.

Table 6-1. Base and I/M System VM minidisk requirements

Type of Site Being Installed	**Base System**	**I/M System**
Primary Site	600,000 blocks 500 cylinders	102,000 blocks 85 cylinders
Secondary Site on the same processor with a backbone copy of the file system	600,000 blocks 500 cylinders	Shared
Secondary Site on the same processor with a secondary copy of the file system	360,000 blocks 300 cylinders	Shared
Secondary Site on a different processor with a backbone copy of the file system	600,000 blocks 500 cylinders	102,000 blocks 85 cylinders
Secondary Site on a different processor with a secondary copy of the file system	360,000 blocks 300 cylinders	102,000 blocks 85 cylinders

2. Determine if the AIX minidisks Totals from the AIX minidisk worksheet(s) are to included on the Base System VM minidisk (go to Step 2a), if they are to be place on their own VM minidisk (go to Step 2b, skip Step 2a), or if they are to be grouped VM minidisks (go to Step 2b, skip Step 2a).

   a. Add all the AIX minidisk Totals from the AIX minidisk worksheet together, then continue to Step 3.

   b. Record how the AIX minidisks are to be placed in the AIX minidisk rows, then continue to Step 3.

3. Convert all AIX minidisk 1024 blocks to FB-512 blocks by:

   Number of blocks * 2 = ***Number of FB-512 blocks***

   If a CKD device is used for a VM minidisk, convert the FB-512 blocks to 3380 cylinders by:

   Number of FB-512 blocks / 1200 = ***Number of 3380 cylinders***

4. If the Base System VM minidisk is to include the AIX minidisks, add the converted number of FB-512 blocks or number of cylinders to it and

record the total in the additional space for the Base System entry in VM minidisk worksheet. Otherwise, record the converted AIX minidisk totals in the proper AIX Minidisk row. If several AIX minidisks are grouped together, combine their totals in the # of FB-512 blocks or # of cylinders row.

5. Record the Volume ID or real address (from your system) for each VM minidisk.

6. Record the Starting block/cylinder (or entire disk) of the disks to be used.

6.4 Planner's Checklist

In addition to planning for AIX and VM minidisks, the planner must consider Transparent Computing Facility (TCF) requirements. Planning a TCF cluster requires many essential steps. Appendix A, "Worksheets" contains a Planner's Checklist which enumerates these important items. The following table provides typical values for a heterogeneous cluster.

If this is your second pass through the material, you should now have developed a full set of plans for a working cluster. Reproduce the Planners Checklist at this point and fill it out. This will alert you to any oversights.

Subtopics

6.4.1 Typical Cluster Requirements

Refer to the following table for typical cluster requirements.

Table 6-2. Typical Values for a Heterogeneous Cluster

Item	**Fixed Disk**	**Number of Blocks**	**Number of Files**
/aix370	0	24000	1500
/ (root)	0	236000	16000
dump	0	6250	
page	0	40000	
/aix370/tmp	0	24000	6664

7.0 Chapter 7. The AIX/370 Guest System Virtual Machine

Subtopics

7.1 Contents

7.2 About This Chapter

This chapter discusses how to set up a virtual machine suitable for an AIX/370 guest system.

7.3 Setting Up an AIX/370 Virtual Machine

When planning for AIX/370, consideration must be given to the virtual machine where AIX/370 is to be initialized. For setting up AIX/370 under VM, these areas must be considered in addition to the information later in the chapter:

How to set up the user-directory

Whether a machine that is capable of XA mode is to run in XA mode o 370 mode.

Whether AIX/370 is to run

- ***V=R*** (virtual=real mode)
- ***V=V*** (virtual=virtual mode)
- ***V=F*** (virtual=fixed mode)?

Note: ***V=F*** machines are only available on 3090 Series E (or later) processors with the 3090 Multiple High Performance Guests Support feature or the Processor Resource/System Manager (PR/SM) feature installed. In addition, if VM/XA SP Release 1 is used, the enhancement for multiple preferred guests which is available on a program update tape (PUT) must be installed.

Whether a VM processor with PR/SM capability is to run the AIX/37 guest system virtual machine in PR/SM mode (in one of the PR/SM partitions) or in normal mode on the main processor.

VM options for AIX/37

For VM options and parameters, check with the VM administrator. For some installations, this may be the same person who acts as the AIX/370 system administrator. For most installations, the VM administrator and the AIX/370 system administrator will be separate, especially if AIX/370 is a guest (among others) running under VM. For information on VM options, refer to the *VM/XA System Product Planning and Administration* and *VM/SP Planning Guide and Reference*.

Which VM performance options to control scheduling algorithms (fo example, the System Resource Manager - SRM). For more information, refer to "General Considerations for the VM Environment," in either the *VM/XA System Product CP Command Reference* or the *VM/SP System Product CP Command Reference*.

Whether to use dedicated disks versus VM minidisks. The thre alternatives are under VM are:

- To dedicate an entire disk pack to AIX usage and let AIX manage the disk space by its own methods (dedicated disk)

- To allocate an entire pack as one minidisk assigned to AIX and let AIX manage the disk space by its own methods (full pack minidisk)

- To allocate some segment of a real pack as one minidisk assigned to AIX; space management is performed by VM at the minidisk level and by AIX within the VM minidisk.

Subtopics

7.3.1 General Considerations for the VM Environment

The following elements should be considered when working in the VM environment.

Each AIX/370 cluster site must be allocated enough service from th underlying VM systems so that it can respond reasonably to all offered network load. Additionally, any AIX/370 cluster site which is relaying packets in order to provide a communication path between two or more other cluster sites must be a system with sufficient computing cycles to forward the packet promptly.

VM Parameter

- Set ASSIST ON (if supported by the hardware and the installed version of VM) for AIX/370 to get full advantage of the VM assist functions if activated for the entire system.

- For AIX/370, use a virtual storage size between the following values:

  -- The possible virtual memory size for ***V=R*** or ***V=F*** is between the following limits. The lower limit is 6Mb. The upper limit is the real machine size minus VM required memory. The VM required memory consists of the sum of the following:

     Memory required for CP-nucleus
     Memory required for free space
     Sufficient memory to run the total number of CMS users expected.

  -- The possible virtual memory size for ***V=V*** is from 6Mb up to 16Mb (for System/370 machines) and from 6Mb up to 999Mb (for System/370XA machines).

     As a general rule, the size of virtual memory should be set to no more than 1.2 to 1.5 times the value of the real storage that can be made available to the virtual machine. This estimate is dependent upon processor speed and the average load being carried. A low-end processor running a heavy load will operate best with a lower virtual-to-real-storage setting. A high-end processor that is lightly loaded and has fast paging devices can probably handle a much higher setting.

7.3.2 Initializing AIX/370

Assuming that VM is initialized, there are five possible ways to start up AIX/370:

The start procedures for AIX/370

- AUTO IPL (Directory IPL statement)
- CMS EXECs.

From the VM Directory, IPL CMS, and from CMS, IPL AIX/370. The tw directory examples which follow illustrate this first method. The second statement in each directory performs the IPL of the appropriate form of CMS.

Once CMS is initialized, this process can be further automated by having the IPL of AIX/370 performed as part of a CMS PROFILE EXEC. Such an EXEC may include the setting of the needed VM options. Other VM set options need to be reset every time AIX/370 is initialized, and these are ordinarily made part of an AIX/370 startup script.

This is an often used option. Its advantage is that CMS is available when needed for performing tasks (for example, formatting disks).

Note: In order to initialize AIX/370 from a CMS EXEC in V=R mode, a special CMS nucleus may have to be built.

From the VM Directory, IPL the device which serves as the AIX/370 Boo Pack. Set the needed VM options manually or make them part of an AIX/370 startup script.

Initialize AIX/370 manually. The VM directory contains no IP statement. Appropriate commands to bring up AIX/370 are given on the VM command line. Further commands are given on the command line from within AIX/370.

To simplify the starting of AIX/370 under VM, consider writing CM EXECs to do the following operations before AIX/370 is initialized:

- Defining and coupling Virtual-Channel-to-Channels
- Attaching devices (for example, LAN device, tapes, and printers) before an IPL of AIX/370

  Note: This requires that the AIX/370 userid have Class B privileges in the VM directory.

- Initializing the AIX/370 Base System VM minidisk. To simplify this EXEC, the devices can be defined in the directory rather than attached when AIX/370 is initialized.

For information about writing EXECs, refer to the following VM documentation:

- *VM/SP System Product Interpreter User's Guide*
- *VM/XA System Product Interpreter User's Guide*
- *VM/SP System Product Interpreter Reference*
- *VM/XA System Product Interpreter Reference*.

Note: The methods outlined above attach devices during the initialization of AIX/370. Devices can also be accessed by defining them in the VM directory; this makes them part of the virtual machine prior to initializing AIX/370.

7.3.3 Coexistence with Other Guests

In a VM environment, AIX/370 can coexist with other guest operating systems, but it is important to remember that the requirements of the other guests will influence the settings that were previously discussed. The planner must take into account the total interaction between VM, AIX/370, and all other guests operating on the processor. For more information on determining these settings, refer to *VM/XA System Product Planning and Administration* and the *VM/SP Planning Guide and Reference*.

7.3.4 Virtual Machine Operating Environment

There are many performance considerations to be addressed when AIX/370 operates under VM. One of the critical concerns in the execution of any operating system is the amount of physical memory available to the system. In the case of VM, this would translate to the amount of physical memory available to the AIX/370 virtual machine.

VM/SP can execute a virtual machine in ***virtual equals real*** (***V=R***) memory mode and ***virtual equals virtual*** (***V=V***) memory mode. VM/XA has an additional memory mode of ***virtual equals fixed*** (***V=F***).

Note: ***V=F*** machines are only available on 3090 Series E processors with the 3090 Multiple High Performance Guests Support feature or the Processor Resource/System Manager (PR/SM) feature installed. In addition, if VM/XA SP Release 1 is used, the enhancement for multiple preferred guests which is available on a program update tape (PUT) must be installed.

One virtual machine at a time on the physical machine can be dedicated as the V=R machine. For this virtual machine, it is possible to disable a number of the translations that are normally necessary for a virtual machine. The disabling of translations can typically be done by setting the **NOTRANS** option.

A VM/SP virtual machine that is not V=R is referred to as V=V; for a VM/XA virtual machine, V=F is another option. There are also performance considerations, like expanded storage support, when using V=V. It is possible in V=V mode to dedicate a particular number of real pages for a particular virtual machine.

On VM/XA machines, V=F mode is another option. V=F machines have some of the characteristics of V=R machines and some of the characteristics of V=V machines. Because of the mix of characteristics, performance for these machines should be better than the same machine running as a V=V machine and may be as good as the same machine running as V=R.

There are a number of performance flags that can be set which impact scheduling. For example, on a multi-processor, it is possible to dedicate a particular processor a particular virtual machine. It is also possible to set priorities and other scheduling parameters by using the appropriate **SET** option. For more information on ***V=R***, ***V=V***, ***V=F***, and performance tuning, refer to the *AIX/370 Administration Guide*, the *VM/SP Planning Guide and Reference*, and *VM/XA System Product Planning and Administration*.

Subtopics

7.3.4.1 The V=R Environment

The V=R environment is the preferred environment for AIX/370.

Advantages

Reduces the CPU time that is needed for management of storage, paging and I/O.
Results in better response time
VM recovery

Disadvantages

Allocates a fixed amount of main storage, independent of the number o users, which is unavailable to other users.
Cannot adjust the allocation of main storage to changing demand because of changing workloads without regenerating and rebooting VM and AIX/370 (for example, loads from CMS and other guests).

General Considerations

Portion of real storage permanently allocated to AIX/37
Can be unlocked when AIX/370 logs off, but can not be reclaime without an IPL of VM
Paging is handled by AIX/370 (and not by VM
No need for shadow table
Exploit VM assists to reduce CPU tim
No double translation of channel command words (CCWs)

Operation for VM/SP

During the CP nucleus generation, prepare the V=R region. (At leas 6Mb, preferably six or more, depending on the expected load.)
Dedicate devices to AIX/370 to avoid CCW translations. **NOTRANS ON** is ignored for devices that can be accessed on alternate paths.
Issue **SET FAVOR** (without percent) after the virtual machine has logged on.
Issue **SET STBYPASS VR** and **SET NOTRANS ON** after an IPL of AIX/370 has been performed in the virtual machine.

Note: The last two SET commands are reset at every IPL of the virtual machine. They can be issued manually, but the usual method is to include them in the AIX/370 startup scripts.

Operation for VM/XA

During the CP nucleus generation, prepare the V=R region large enoug for V=R and V=F machines.
Dedicate devices to the V=R machine to avoid CCW translations
Dedicate a processor to the V=R machine if desired (this only applie to a multi-processor configuration).
Issue **SET NOTRANS ON** after an IPL of AIX/370 has been performed in the virtual machine.

7.3.4.2 The V=V Environment

The V=V environment should be used for testing, casual or moderate use of AIX/370. This is the mode to use for AIX/370 if some other operating system is being selected as a preferred guest (such as MVS).

Advantages

Real storage can be shared with other application
Use of real storage above 16M

Disadvantages

Higher cost in CPU cycles for management of resource
Increase of response tim
May require complex trial-and-error tuning

General Considerations

Main storage is managed by both, AIX/370 and V
Needs shadow tabl
System does double CCW translatio
May result in double pagin
Needs more CPU time to manage resource
May have higher response time

Operation
Issue VM SET commands (as needed) after the logon of AIX/370.

7.3.4.3 The V=F Environment

Advantages

Reduces the CPU time that is needed for management of storage, paging and I/O
Results in better response time

Disadvantages

Requires additional fixed storage in the V=R area, which become unavailable to other users
Cannot adjust the allocation to changing demands without restartin AIX/370
Does not have the recovery facilities of V=R
Requires VM/XA SP and PR/SM hardware

General Considerations

Portion of real storage permanently allocated to AIX/37
Can be used by other machines defined as V=F, but only if AIX/370 i logged off
Paging is handled by AIX/370 (and not by VM/XA
Shadow tables are used, but not frequently update
Exploit VM assists to reduce CPU time

Operation for VM/XA

During the CP nucleus generation, prepare the V=R region large enoug for V=R and V=F machines
Dedicate a processor to AIX/370 if enough resources are availabl (this only applies to a multi-processor installation).

7.3.5 Directory of AIX/370

Figure 7-1 and Figure 7-2 show typical VM directories for the setup of two different AIX/370 systems. Figure 7-1 is a typical small-sized AIX/370 system running on a low-end processor. Figure 7-2 is a typical large-sized AIX/370 system running on a high-end processor. In both figures, the right column is comprised of comments which may not actually appear on the display. The note references (**See Note #**) in the comments section of both figures are listed on page 7.3.5.

```
USER AIX370 password 6MB 16M G              - Userid AIX370
IPL CMS                                     - IPL CMS
ACCOUNT 12345678                            - Accounting
OPTION ECMODE BMX V=R                       - ECMODE ON, V=R is optional
CONSOLE 500 3215 A OPERATOR                 - See Note 1
LINK userid 190 190 RR                      - See Note 2
LINK userid 19D 19D RR                      - See Note 2
LINK userid 19E 19E RR                      - See Note 2
MDISK 191 FB-512 84700 1000 PERF21 M U U    - Define CMS minidisk
SPOOL 00A 1403 x                            - See Note 3
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH  A
SPOOL 00E 1403 A
DEDICATE 320 xxx                            - Dedicate all the disks to be
DEDICATE 340 xxx                            - assigned to AIX/370 and/or
DEDICATE 341 xxx                            - specify VM minidisks
DEDICATE 342 xxx                            - as desired. See Note 4.
DEDICATE 344 xxx                            - xxx defines the real
DEDICATE 345 xxx                            - address of the AIX/370 volume
MDISK 360 FB-512 85700 400000               - Defines VM minidisk for AIX/3
DEDICATE 160 xxu                            - Dedicate 8232 LAN device, See
DEDICATE 161 xxv                            - Dedicate 8232 LAN device
DEDICATE 100 xxw                            - Dedicate CETI LAN device, See
DEDICATE 101 xxx                            - Dedicate CETI LAN device
DEDICATE 102 xxy                            - Dedicate CETI LAN device
DEDICATE 103 xxz                            - Dedicate CETI LAN device
```

Figure 7-1. Directory of a Small-sized AIX/370 System Running under VM/SP

```
*
USER AIX370-1 password 16M 256M G                       - Userid AIX37¦-1
IPL CMSXA                                               - IPL CMS in X¦ mod
ACCOUNT 12345678                                        - Accounting  ¦
OPTION V=R                                              - Optional    ¦
MACHINE XA                                                            ¦
CONSOLE 500 3215 T OPERATOR                             - See Note 1  ¦
CPU 1 CPUID 073715                                      - Multiple Pro¦esso
LINK userid 0190 0190 RR                                - See Note 2  ¦
LINK userid 019D 019D RR                                - See Note 2  ¦
LINK userid 019E 019E RR                                - See Note 2  ¦
LINK userid 031A 031A RR                                              ¦
MDISK 0191 3380 1645 0002 KOM32B MW rpass wpass mpass   - Define CMS m¦nidi
```

```
¦   MDISK 0390 3380 0001 0380 KOM32B MW rpass wpass mpass  - Define AIX m¦nidi
¦   MDISK 0391 3380 0381 0442 KOM32B MW rpass wpass mpass  - Define AIX m¦nidi
¦   SPOOL 00A 1403 x                                       - See Note 3  ¦
¦   SPOOL 00C 2540 READER *                                              ¦
¦   SPOOL 00D 2540 PUNCH  A                                              ¦
¦   SPOOL 00E 1403 A                                                     ¦
¦   SPOOL 01E 1403 A                                                     ¦
¦   SPECIAL 0140 CTCA     AIX370-2                         - See Note 7  ¦
¦   DEDICATE 0392 0480                                     - Dedicate all¦the
¦   DEDICATE 0393 0481                                     - assigned to ¦IX/3
¦   DEDICATE 0394 0482                                     - specify VM m¦nidi
¦   DEDICATE 0395 0483                                     - as desired. ¦ee N
¦   DEDICATE 0160 xxu                                      - Dedicate 823¦ LAN
¦   DEDICATE 0161 xxv                                      - Dedicate 823¦ LAN
¦   DEDICATE 0100 xxw                                      - Dedicate CET¦ LAN
¦   DEDICATE 0101 xxx                                      - Dedicate CET¦ LAN
¦   DEDICATE 0102 xxy                                      - Dedicate CET¦ LAN
¦   DEDICATE 0103 xxz                                      - Dedicate CET¦ LAN
¦                                                                        ¦
¦                                                                        ¦
+------------------------------------------------------------------------+
```

Figure 7-2. Directory of a Large-sized AIX/370 System Running under VM/XA SP

Notes:

The following notes refer to Figure 7-1 and Figure 7-2.

1. It is customary to use **OPERATOR** as the userid for the VM system operator, although other userids can be used. **OPERATOR** can be defined as a secondary user of AIX/370. This makes **OPERATOR** a secondary user of the AIX/370 guest system virtual machine and it eases communication with AIX/370 when:

   a printer is closed
   tapes or disks are detached
   attached devices are queried.

   The secondary user facility can only be used if AIX/370 is disconnected from its logon terminal after it is initialized.

2. **USERID** (in the **LINK** statement) is the user-identifier of the VM administrator (usually, this identifier is MAINT). Add links for all VM system disks.

3. Add this **SPOOL** statement for console logging if a class other than **A** is desired. The **x** should be substituted with the desired class.

4. Add **DEDICATE** statements for disks or tapes as needed. **ATTACH** statements can be used instead. For example, the following command is used to attach the tape to AIX/370:

   ```
   ATTACH xyy AIX370 380
   ```

   where **xyy** is the real address of the tape drive. For DASD allocation, **MDISK** statements can also be used. Depending on the installation, any mixture of these statements may be used when assigning disk devices.

   The **DEDICATE** statements for the LAN-attached devices match two columns of addresses. The VM system administrator will provide one column and the AIX administrator will provide the other. They must work together to provide the needed match.

5. Add **DEDICATE** statements for the 8232 LAN device if used. The **xxu** and **xxv** refer to real addresses which increment by 1, consecutively.

   Note: Each network interface within the 8232 requires two VM device addresses.

6. Add **DEDICATE** statements for the CETI LAN device if used; the integrated LAN adapters of the 9370 are CETI LAN devices. The **xxw**, **xxx**, **xxy**, and **xxz** refer to real addresses which increment by 1, consecutively. One LAN device uses four addresses.

7. This statement allows a virtual channel-to-channel connection between **AIX370-1** and **AIX370-2**, which are running on two different virtual machines in the same processor. Either side must issue a CP COUPLE command before the VCTCA can be used. If AIX/370 is executing on two separate real machines, a CTCA (3088) may be dedicated to each end instead of the VCTCA.

8. The statement for multiple processors is optional.

8.0 Chapter 8. Customization Planning

Subtopics

8.1 Contents

8.2 About This Chapter

Certain parts of the AIX/370 Operating System can be customized to suit the needs and tasks of a particular installation. This chapter presents customization details for shell scripts, system parameters, and applications as well as information on transferring programs and device drivers.

8.3 What Can Be Customized

AIX/370 and AIX PS/2 are distributed as object code only (OCO). That is, no AIX source code is shipped. Therefore, AIX commands and routines that are written in C language or assembly language can not be customized except as provided for in the distribution software.

Although the AIX/370 is received as an object code only system, the following can be customized:

Shell script
System parameter
Applications

In addition, the TCF cluster may be customized by adding your own programs and device drivers to it.

Subtopics

8.3.1 Shell Scripts

Some of the AIX/370 commands and procedures are distributed as shell scripts for example, the startup and shutdown procedures.

Note: Before the AIX shell scripts are changed, save the original version as backup. It can happen that program temporary fixes (PTFs) overlay certain shell scripts that have been customized. Therefore, check the PTFs to see whether they overlay customized shell scripts. If they do overlay, make sure that the changes are re-applied to the shell script after the PTF application. For more information on PTFs, refer to "Preventive Service for AIX/370" in topic 13.3.1.

8.3.2 System Parameters

Important system parameters can be changed, for example:

Buffer size **buffer**)
Number of mount-table entries for the local cluster site **mount**)
Number of global mount-table entries for the entire cluster **gmount**)
Number of process-table entries **procs**).

For more information on these parameters, refer to *Managing the AIX Operating System*.

8.3.3 Applications

Applications can be customized by making appropriate changes in the corresponding configuration files. Some examples of applications are:

cc	TCP/IP	printer	NFS
mail programs	INed	INnet	INmail
accounting	uucp		

The exact location of the configuration files for each these applications is to be found in the appropriate AIX manual.

8.3.4 Transferring Programs

Programs written in C language can be ported to the TCF cluster. Your own commands and applications can also be added to the TCF cluster. For more information, refer to Chapter 9, "Program Compatibility Considerations."

8.3.5 Device Drivers

Although AIX/370 is received as an object code only system, your own device drivers can be added to the kernel (AIX/370 or AIX PS/2) of the site(s) where the device is to function. For a description of interfaces and system routines that can be used, refer to the "Device Drivers" appendix in the *AIX Operating System Technical Reference*.

9.0 Chapter 9. Program Compatibility Considerations

Subtopics

9.1 Contents

9.2 About This Chapter

Planning for the AIX/370 Operating System may involve the adaptation of programs originally written for a different operating system. Programs written in certain languages can be easily ported to the AIX/370 environment. There should be virtually no problems in porting UNIX System V applications to AIX if the applications are written in C language. However, some of the application programs may need code changes in certain areas. Porting support is provided on source code only.

There is high compatibility between AIX, UNIX System V, and 4.3BSD. This chapter discusses the porting of programs written in C, Assembler, and FORTRAN.

9.3 Differences between UNIX System V and the TCF Cluster

An AIX/370 or AIX PS/2 system contains changes, deletions, and additions which affect certain UNIX System V and 4.3BSD programs. For information about these changes, refer to the *AIX Operating System Commands Reference* and the *AIX Operating System Technical Reference*.

9.4 Porting C Programs to the AIX Platform

The following may be required to transfer (or port) C code from UNIX System V or 4.3BSD to the AIX platform:

Compile the code on the AIX system. If the resultant program compile without errors and/or warnings, the process is complete.

If errors and/or warnings occur, perform the following steps

1. Attempt to debug the program using standard techniques:
   a. Address all errors and/or warnings produced by the compiler
   b. Use the **lint** debugger (the **-p** flag will check for portability problems)
2. If standard debug procedures fail to resolve the problems, suspect portability problems and perform the following:
   a. Recompile the C code. Recompile the code on the original system (that is, on the non-AIX system from which the program is being ported). This recompilation may produce warning flags. Such warnings may represent problems that were minor on the original system and were allowed to go unattended. These unattended problems may not have affected the execution of the code on the non-AIX system, however, they may be preventing the code from executing on the AIX system. The AIX compiler has very strict requirements.
   b. Use **lint -p** to detect other faults that may lead to portability problems.
   c. Resolve any portability problems that may have been found by changing the source code. For example, the size of constants may have to be changed, since they are often processor-dependent.

The following section discusses the C compiler and the main differences between UNIX System V and the TCF cluster. Use this information when executing the above steps.

Subtopics

9.4.1 AIX/370 C Compiler

The AIX/370 C compiler:

- Carries out extensive type checkin
- Supports ANSI standard
- Strictly enforces the rules for variable declaration and typ checking.

The syntax for the ***assignment operator*** is:

```
<operator>=
```

instead of:

```
=<operator>
```

9.4.2 C-Language Portability Considerations

The following are rules and hints for porting programs from other UNIX System V compatible systems to the TCF cluster. This list includes general C portability issues.

Integer *int*) variables and constants are 32 bits for AIX/RT, AIX PS/2, and AIX/370.

De-referencing NULL (with read, write, or execution accesses) i illegal. In user programs, a segmentation violation may occur. In Kernel mode, a Kernel protection mode violation PANIC condition may occur.

De-referencing a pointer or an array out of the bounds of th pointed-to structure or array is illegal and not portable. The results may be the same as de-referencing null. Statements of the form:

```
for (cp = &array[0]; cp < &array[SIZARRAY]; cp++)
    myfun(cp);
```

are valid because there is no de-referencing implied by **&array[SIZARRAY]**.

Memory allocation functions may not allocate the memory that wa assumed. In particular, the **sbrk()** function may not return memory contiguous with the previous allocations. The **malloc()** function is preferred to the **sbrk()** function for memory allocation, since the **malloc()** function takes an unsigned integer parameter and the **sbrk()** function takes a signed integer.

Bitfields are not implemented in all C compilers. When they are, n field may be longer than an integer, and no field can overlap an integer boundary. If necessary, the compiler will leave gaps and move the field to the next boundary. Also, bitfield position is machine dependent. Bitfields in unions cannot be expected to port at all.

Pointers are special in C. Unlike other variables, only adding to an subtracting from them is allowed; even then, the addition or subtraction must be done inside a data object. Do not assume that text addresses are even.

No implicit assumption should be made that two data elements declare and initialized adjacently in code will end up being adjacent in memory. Also, the address may not necessarily be comparable to (that is, greater than or less than) any other address.

Memory may not be contiguous, in that physical memory may b discontiguous, and setting a pointer to zero and repeatedly incrementing it may not be legal and may not result in scanning all of memory.

The left shift operator **<<**) is a logical shift. This means that it moves zeroes into the vacated bits. The right shift operator (**>>**) when applied to an unsigned quantity is a logical shift. When applied to a signed quantity, the vacated bits may be filled with the sign bit or may be filled with 0. To assure consistent and portable results, mask out the offending bits, for example:

```
char c;
c = ( c >> 3 ) & 0x1f;
```

Constants used for computation are integers unless they are cast t long. The results of arithmetic with integer constants are also integers unless at least one of the components is as long.

Programs written for AIX/370 Version 1.2.1 Japanese locales must be written according to a particular codeset-independent technique. This is especially true for programs which will run in a mixed-language environment where input may be single-byte ASCII characters or multibyte characters. For more information on programming for Japanese locales, refer to *AIX Programming Tools and Interfaces*.

9.4.3 AIX Boundary Alignment

The C compilers available for some computer architectures require that variables, arrays, and buffers be aligned on byte, word, or double-word boundaries. This restriction is imposed by the requirements of the processor which will be performing the compilation.

The processor itself does not actually require any alignment from the programmer. Each processor will automatically align the variables given to it, but this takes a certain amount of processor overhead. Code will run faster and with greater efficiency if alignment is performed by the programmer.

For AIX, one of two processors will be involved:

AIX PS/2 - The PS/2 processor imposes very simple limitations. *Int* variables, which are four bytes long, should be aligned on double-word boundaries.

AIX/370 - The System/370 machines impose various limitations. Th compiler usually handles alignment efficiently, but the following areas must be monitored:

- A CCW must start on an 8-byte boundary
- An IDAW must start on a 4-byte boundary
- An *Int* variable must start on a 4-byte boundary
- A *Short* variable must start on a 2-byte boundary
- Each instruction must start on a 2-byte boundary.

The compiler takes care of these alignments automatically when each such entity is declared alone, but, when they are combined into a data structure, the compiler has no way to recognize them.

A structure will always start properly aligned, but any of the above areas that are declared within the structure are the responsibility of the programmer to align. In most cases, the only effect of failing to do so is a significant reduction in processing efficiency. However, failing to properly align a CCW will cause a system failure.

For more general information on boundary alignment, refer to the *IBM System/370 Extended Architecture Principles of Operation* and *IBM System/370 Principles of Operation*. For more specifics of C language usage, refer to the *AIX C Language Guide* and *AIX C Language Reference*.

9.4.4 External Variables for AIX/370

An external variable should be defined only once among all the functions and files that make up the source program, and all declarations of that variable must be marked external.

The following points and example pertain to variable definitions and external declarations:

If a variable is defined in **File 1** (see Figure 9-1) and is used in **File 2** (see Figure 9-2), **File 2** must have an external declaration of that variable to connect the two occurrences of the variable.

An external variable must be declared external in each function tha accesses the variable.

This external declaration is not mandatory if the definition of the variable precedes the function definition in the same source file, or if a global external declaration (for an external variable that is defined in another file) precedes the function definition in the same source file.

It may be helpful to keep a single copy of common externa declarations in a header file which can be inserted in each source file where it is needed using the **#include** file inclusion facility.

The following code (refer to Figure 9-1 and Figure 9-2) illustrates some of the proper uses of external declarations.

```
int x, z;
funct1()
{
    ...
    x = 7;          A
    ...
}
```

Figure 9-1. File 1

```
extern int x;       B
funct2()
{
    ...
    extern int z;   C
    extern int y;   D
    z = 14;
    y = 21;
    x = 8;          E
    ...
}
int y;
```

Figure 9-2. File 2

Notes:

A The definition of external variable **x** precedes the definition of **funct1**. Therefore, an external declaration of **x** is not required.

B **x** is defined in another file.

C **z** is defined in another file.

D The definition of external variable **y** follows the definition of **funct2** in the same file. Therefore, an external declaration of **y** is required within **funct2**.

E A global external declaration of **x** precedes the definition of **funct2**. Therefore, an external declaration of **x** is not required within **funct2**.

9.4.5 Block Size

The block size in both AIX/370 and AIX PS/2 is 4096 bytes. In the TCF cluster, the system header file **/usr/include/sys/param.h** has a defined constant BSIZE which has the correct size for AIX/370 and AIX PS/2.

Corresponding constants should be changed in all programs and utilities which have to know the block size to use the constant. Writing in multiples of BSIZE results in performance improvements over writing in other sizes.

9.4.6 Data Type Size

For AIX/370, AIX PS/2, and AIX/RT, the sizes of basic C data types are:

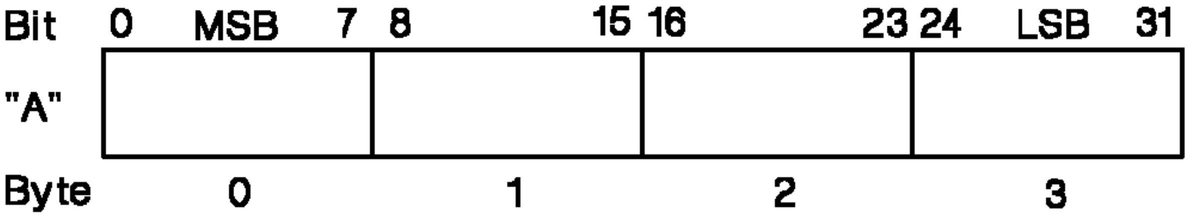
Data type	**Bits**
char	8
int	32
short	16
long	32
float	32
double	64

9.4.7 Bit and Byte Ordering

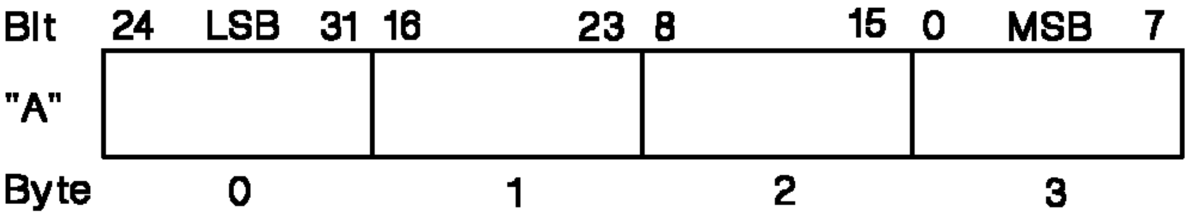
Bits within a data type are numbered from left to right. The presentation of (signed) integers is as follows:

Bit 0 is the sign bi
Bits start at address
Digits increase in significance from bit 31 to 0

A 32-bit (signed) integer in AIX/370 has the following format:



A 32-bit (signed) integer in AIX PS/2 has the following format:



9.4.8 Floating Point Representation

AIX/370 floating point numbers can be represented in 32, 64, or 128 bits. The exponent is 7 bits long and is expressed in ***excess 64 notation***. The radix is 16, rather than 2; this produces an approximate range of: 16(-64) to 16(63), or 10(-79) to 10(75).

AIX PS/2 floating point numbers can be represented in 32 bits (float) or 64 bits (double). Floats are stored with 8 bits of exponent (biased by 127), 23 bits of mantissa, and 1 sign bit. The approximate range of floats is 8.43x10(-37) to 3.37x10(38). Doubles are stored with 11 bits of exponent (biased by 1023), 52 bits of mantissa, and 1 sign bit. The approximate range of doubles is 4.19x10(-307) to 1.67x10(308).

9.4.9 Sign Extension

AIX PS/2 and AIX/370 do not produce a sign extension to a negative integer when *char* is converted to *int*.

9.4.10 Files

Changes have been made to the constants, structures, and definitions in several include files in **/usr/include/sys.** Users who use the information in these files should first consult the AIX version of these files.

9.4.11 File Naming Conventions

AIX follows the standard convention for naming files. The following suffixes should be used:

Suffix	**File**
c	C language
h	header
s	assembly language
f	FORTRAN
o	object

9.4.12 Clock Location and Supervisor Area

With AIX/370, it is not possible to directly read the clock location or the supervisor area. All such information must be accessed through the appropriate system call.

9.4.13 Preprocessor Name

The preprocessor symbol for AIX/370 is **u370**. The preprocessor symbol for AIX PS/2 is **i386**.

9.4.14 Differences in System Calls and Subroutines

The **lockf** system call has been changed to reflect the TCF cluster environment. The deadlock detection is only done at the local site, so multi-site access of a file can possibly cause deadlock. If the file is replicated, only the primary copy can be locked. If a file is locked and the primary site of that file stops operating, AIX sends your process a SIGPIPE signal.

The **stat** system call returns a global file system number in **st_dev** rather than a major/minor device number. In general, comparisons between **st_dev** and **st_rdev** fail.

The first argument of the **ustat** system call is a global file system number rather than a major/minor device number. The changes to both **st_dev** and **ustat** are consistent when used in conjunction. The correct information is obtained when they are used.

The existing **ttyslot** subroutine does not function properly when executed at a cluster site distinct from where the terminal (tty) is connected.

Since the access and modification times of a file result from th storage site clock value, programs expecting to do comparisons between these times and the result of **time()**, should expect to see a deviation dependent upon the slight variations of the clocks on the machines in the cluster.

Since AIX provides an atomic **commit** mechanism, overwriting a file (that is, not truncating the file first) results in new data pages being allocated to the file in replacement of the old ones. Programs depending on physical data blocks being at the same location do not function properly in the AIX environment.

9.4.15 Transferring AIX Files

AIX/370 File Transfer: The tapes used to transfer files between two AIX/370 systems may be written at 1600 or 6250 bpi density. Files from other UNIX System V or 4.3BSD systems to AIX/370 systems should be written at 1600 or 6250 bpi density.

Because AIX/370 has a different internal inode-structure and uses the block size of 4096 bytes, any file transfer should be done by using the **cpio** or **tar** utilities; do not use the utilities that generate binary images to transfer file systems between AIX/370 and other UNIX System V compatible systems.

For more information on inodes, refer to *Managing the AIX Operating System*. For more information file formats, refer to the *AIX Operating System Technical Reference*.

AIX PS/2 File Transfer: To transfer file systems from AIX/RT to AIX PS/2, copy them onto diskettes or streaming tape using the **tar, cpio,** or **backup** command and read them in using the appropriate command under AIX PS/2.

Note: When moving files between UNIX System V systems and any AIX systems, use **cpio** with the **-c** flag. This copies the file with ASCII headers for portability.

9.5 AIX/370 Assembler Restrictions

The following restrictions apply to the AIX/370 Assembler.

Subtopics

9.5.1 Assembly Language Routines

Routines written in a machine language must be rewritten in AIX/370 Assembler or C language.

The following explains the restrictions that apply to the AIX/370 Assembler as compared with the System/370 Assembler Language (also known as Basic Assembler Language or BAL).

9.5.2 Restrictions in Pseudo-Operations

The following pseudo-operations are not recognized:

COPY	PUNCH
START	REPRO
COM	OPSYN
DXD	PUSH
CXD	POP
ICTL	ACTR
ISEQ	

The following pseudo-operations are tolerated (the operations are recognized, but no action is taken):

PRINT	TITLE
EJECT	SPACE

The following pseudo-operations have restrictions as follows:

EQU	The optional length and type arguments are ignored.
DC	Modifiers S and E are not supported.
	Types L, P, Z, S, and Q are not supported.
MEND	May not contain a sequence symbol.
MNOTE	The severity number, if present, is ignored.
WXTRN	Is treated as EXTRN.

9.5.3 Unsupported Attributes

Unsupported attributes are: S' and I'.

Partially supported attributes are:

T', N', and K' may only be applied to variable symbols (&-symbols)

T' (type) and L' (length) attributes, when used in conditiona assembly statements, may only be applied to symbols previously encountered.

9.5.4 Assembler Symbols

Arrays of **Set Symbols** are not allowed. The only **System Variable Symbols** supported are:

&SYSECT
&SYSLIST
&SYSNDX

9.5.5 Macros

An operation field within a macro may not contain a variable symbol.

9.5.6 Conditional Assembly Instructions

Branching backwards via the conditional assembly instructions AGO and AIF is not permitted in open code.

9.6 AIX/370 FORTRAN

AIX offers a VS FORTRAN Version 2 compiler, which produces highly optimized code. For more information on FORTRAN, refer to "FORTRAN" in topic 10.3.2.

10.0 Chapter 10. Application Programming Planning

Subtopics

10.1 Contents

10.2 About This Chapter

When planning for the AIX/370 system, allow for the development of new, custom applications. This chapter describes the application program development tools that are available.

10.3 High-Level Languages Supported

The following section discusses the high-level languages that are supported in the AIX/370 environment.

Subtopics

10.3.1 C Language

C language can be used for structured programming. C is useful for writing operating systems; for example, most of AIX is written in this language. Furthermore, the language can be used to write machine-independent application programs, especially if used with the C program checker **lint**.

10.3.2 FORTRAN

FORTRAN is a mathematically-oriented, high-level, procedural language used to solve numerical problems. The FORTRAN language supported by VS FORTRAN Version 2, including its mathematical function provisions, is compatible with the standards language for FORTRAN, ANSI X3.9-1978 and ISO 1539-1980. VS FORTRAN Version 2 also provides compatibility with programs written using FORTRAN language X3.9-1966 and supports many IBM extensions to FORTRAN-77.

The VS FORTRAN Version 2 product is a full-functioned offering that includes support for the 3090 vector facility. It has the ability to perform vector and standard sequential processing within a single program.

AIX/370 includes a VS FORTRAN Version 2 compiler.

10.3.3 VS Pascal

VS Pascal is only available on AIX PS/2.

10.4 Tools for Application Programming

AIX offers a set of tools for program development. Most of these tools are contained in the Application Development Toolkit, which is a separate Licensed Program Product. Some of the important components of the Application Development Toolkit are:

admin Used to enter a file into the SCCS system and give it a description which will be stored by SCCS.

as Used to process assembly language statements into machine language.

bs Compiles a language that offers rapid development features drawn from C, BASIC, and SNOBOL.

cc Compiles standard C language code.

cdc Used to change the comments and modification requests in an SCCS delta file (refer to **delta**).

comb Combines SCCS files into composite modules and changes the history files to reflect the merger.

ctags Used in conjunction with the vi and ex editors to create files of tags that are used to rapidly find tagged objects within specified text files. This allows rapid updating of functions within C, VS Pascal, or FORTRAN source code files.

dbx Used to examine object and core files. A symbolic debugger for use with C and/or VS Pascal programs. Provides a controlled environment for running programs.

delta Used to incorporate editing changes made to a source code file and to establish the new, edited version as a new SCCS file.

dump Used to dump specific parts of an object file. Functions similar to the UNIX System V **dump** command.

dumpbsd Works the same as **dump**, but in a fashion characteristic of the 4.3BSD version.

get Gets a SCCS file so that it may be read or edited.

indent Used to format C source code files in many different styles.

ld Combines object files into one file.

lint Checks C code for syntax errors and other problems.

lorder Checks the object files stored in an archive or library and lists all their external identifiers.

m4 Used to perform macro processing so that repeated functions may be stored as simple codes.

make Keeps track of the sequence of steps necessary to compile a program which is composed of many different modules; traces patterns of dependency so that, when a change is made to one module, all the modules dependent on it can be changed in a corresponding fashion.

mkstr Used to create files of error messages, thus reducing system overhead in the execution of programs with large numbers of error messages.

nm Used to print a table of the names or symbols used in a program.

prs Used to print any selected segment of an SCCS file.

rdump A file system dump that works across a network; copies all files changed after a certain date.

regcmp Used to compile Regular Expressions, of the sort handled by the **ed**, **sed** and **grep** commands.

restorebsd Used to read tapes dumped with the **dumpbsd** command.

rmdel Used to remove one of the versions of an SCCS file.

rrestorebsd Used to restore file system(s) across a network. Restores files dumped with **rdump** or **dumpbsd**.

sact Used to print a record of which SCCS files are currently checked out for editing.

sccshelp Provides help within the SCCS program.

shlib2 Used to create a shared library from a set of unshared object or archive files (for PS/2 only).

size Used to find the size of an object file.

strip Used to strip an object file of all symbolic references.

symorder Rearranges a list of names (symbols) into a new order. This program was designed to decrease the overhead of getting symbols from the kernel.

sysdump(370) Initiates various kinds of dumps. Often used to take a ***snapshot*** of a certain section of memory in order to diagnose problems.

unget The reverse of the **get** command; informs SCCS that a file which was removed for editing will not be edited.

unifdef Used to remove from a source code file those lines which begin with "ifdef". The rest of the file is not affected.

val Used to validate an SCCS file; checks for proper version number and other identifying information.

vc Used for Version Control in SCCS tracking operations.

what Used to search SCCS files for a particular module, which is recognized by identifier comments that have been placed in the text.

xstr Used to form and maintain a file called **strings** which contains often-used text strings from many large programs. These shared strings are stored in hashed form for rapid access.

For more information on the tools available for the development of application programs, refer to *AIX Programming Tools and Interfaces* and the *AIX Operating System Commands Reference*.

10.5 Installing Programs

When a new application program is finished, it is often installed on systems other than the one on which it was developed. From time to time, old programs need to be modified and/or updated. AIX simplifies and streamlines these procedures by automating them with the **installp** and **updatep** commands. Only a member of the system group or superuser can use these commands.

In addition to providing an easy-to-interface for the person installing the program or update, installation and update services also help ensure that the installation is correct by:

Maintaining an accurate record of the revision state of the program o the system

Checking the revision level of other needed programs to ensure tha they will work with the program being installed

Providing instructions to install the progra

Providing online copies of changes to documentation

Subtopics

10.5.1 Program Installation Requirements

The primary site (where the primary copy of the replicated root file system is stored) must be operational. The **installp** command will need to make changes to the files in the primary copy of the root system, so that those changes may later be propagated to all cluster sites.

The following is required to perform the installation:

A full set of control files for controlling library function

A full set of program files (the executable modules)

These files must be installed on the installation medium using the **backup -i** command (backup by individual name) before using the **installp** and **updatep** commands. For full details on the **installp** and **updatep** procedures, refer to *AIX Programming Tools and Interfaces* and *Managing the AIX Operating System*

11.0 Chapter 11. Operation Planning

Subtopics

11.1 Contents

11.2 About This Chapter

In planning for the AIX/370 system, consider what operations need to be known and what duties have to be performed by the staff. This chapter describes procedures that will ensure efficient use of resources and aids in the decision about which procedures should be automated. It also covers what information system administrators need to collect and the procedures used to recover from problems that may arise.

A well prepared operation plan ensures the efficient use of resources and helps to reduce problems during operation to a minimum. In planning for the operation task, consider the following:

- What resources will be needed
- What is the relationship between resources (for example, names an addresses)?
- What procedures protect against loss of resources
- What procedures can be used to control the operation of an AIX/370 an AIX PS/2. cluster?
- What options can be used to check the status of program execution
- What abnormal events may occur, and what procedures can be used t recover from abnormal events?

11.3 Considerations for System Administrators

The system administrator of AIX/370 and AIX PS/2 systems must be familiar with the following concepts:

What file systems are (refer to the *Managing the AIX Operating System*).

How to use file protection (refer to the *Using the AIX Operating System*).

A wide range of general user commands, such as

cp
ls.

Distributed processing commands, such as

fast	**on**
fastsite	**ptn**
loads	**site**.
migrate	

Many of the system administration commands, such as

fsck	**minidisks**
fsdb	**mount**.

For more information on these commands, refer to the *AIX Operating System Commands Reference*.

The system administrator must understand how the peripheral devices work. Peripheral devices include terminals, PC/AT and PC/XT workstations running DOS using AIX Access for DOS Users and PCs emulating terminals. For more information, refer to the users guide for the appropriate package.

The system administrator must be aware of:

Who can be logged into what computer in the cluster and how that use is going to use the system. This pertains to privileges and login issues.

System startu

- How file system integrity checking (**fsck**) is done?
- How the initialization sequence (**inittab**) is tailored?
- Which processes are to run and when?
- When cluster communication should be initialized?

System shutdow

- When and how system down-time is scheduled (**/etc/motd** and **wall**)?
- How the system is shutdown (**shutdown**)?
- When cluster communication should be disabled?

Personal Computer use, configuration, and management

- Defining the environment
- Setting up and using file protection.

Tailoring the environmen

- Adapting prototype files that control the environment to the needs of the installation
- Understanding performance tuning.

11.4 Loading, Starting, Restarting, Stopping AIX/370

Because AIX/370 runs as a guest under VM, consideration must be give to the VM environment. The following issues must be considered:

- Should the system administrator start AIX/370 by using CMS procedures?
- Should the system administrator start AIX/370 by using AIX procedures?

AIX/370 provides several procedures to automate

- Starting and stopping of accounting
- Starting and stopping the system activity package
- Device on/off for devices.

These procedures can be changed to suit the requirements of the installation.

11.5 Recording Information on the TCF Cluster

Consider all the information that is needed at the installation and decide which information must be recorded. For example:

Console log For trouble shooting. The log is produced automatically as a VM spool file on AIX/370 as well as being placed in **/usr/adm/messages**.

Accounting information For looking at user accounting data. The information is recorded by the TCF Cluster Accounting System.

System activity For getting information about system load. The information is recorded by the AIX/370 Activity Package.

CPEREP on VM For getting information about hardware and software errors.

Decide if information collecting is to be automated, or if it is to be performed by the system administrator. Data can be collected two ways:

The **cron** command automatically collects data in fixed intervals.

Data can also be collected manually in shell scripts that are used b the system administrator.

11.6 System Accounting

The AIX Accounting System provides methods to:

- Collect resource utilization data
- Record connect sessions
- Monitor disk utilization
- Charge fees to specific logins

If the AIX Accounting System is to be installed and used at your installation, consider the following:

- Which type of accounting records are needed
- Which accounting procedure should be run
- How often must records be collected
- At what times must the procedures run to collect records
- Is there enough space on the file systems to accommodate the wante accounting information; is more disk space required?
- When should the accounting files be cleaned up to free space on disks
- Should system accounting be automated? If yes, how should the cro environment be set up? Should the startup/shutdown procedures be changed to turn system accounting on and off?

Pass answers to these questions to the system administrator for further action. The administrator can customize the AIX Accounting System by changing the setup files that are shipped with AIX/370.

A set of commands and shell procedures are provided to process this accounting data into summary files and reports. For more information about the functions of system accounting, refer to *Managing the AIX Operating System*.

11.7 Status Information on Facilities for TCF cluster

For various purposes, personnel at the installation may need information about program and data status. The following commands can be used in the system administrator environment to obtain this information.

Subtopics

11.7.1 Program Status

Command	Function
acct/*	Obtains accounting information.
crash	Examines the internal structure of an operating system.
lnetstat	Provides statistics about the cluster.
loads	Lists sites that are currently active in the cluster and their workloads.
mount	Shows what file systems are mounted.
ps	Lists process status.
pstat	Provides information contained in certain system tables.
ptn	Lists the current sites in the cluster.
sar	Obtains system load information on AIX/370.
site	Shows the site the user is currently logged into.
uname	Provides information about the current site.
who	Lists logged-in users.

The system administrator can get snapshots of almost all data that is needed by using the **cron** command. This information can be printed or put into a file.

11.7.2 Data Status

Command	Function
df	Shows the state of the file system including blocks and inode usage.
du	Shows disk usage.

11.8 Problems in Operation

When planning for the operation environment, be aware of the problems that may occur. Then, decide on the procedures the operation staff has to follow to recover from a problem. For problems that may occur, refer to *Managing the AIX Operating System* and the *AIX/370 Diagnosis Guide*. Personnel responsible for system operation should also be familiar with the contents of these manuals as well as *AIX Operating System Messages*.

To recover from system problems, the system administrator should be familiar with the tools described in the *AIX/370 Diagnosis Guide*, and should know how to use the VM commands **CP TRACE** and **CPEREP**.

For the convenience of the VM system administrator, consider providing a set of CMS EXECs or PF key settings for entering predefined commands.

11.9 Using The Installation/Maintenance System To Recover From Problems

The Installation/Maintenance (I/M) System can be booted and used to do emergency maintenance on the system. It can also be used to rearrange the Base System volume should performance requirements suggest such a change. Consult *Installing and Customizing the AIX/370 Operating System* for directions on booting the I/M system.

11.10 Coredump Generation and Automatic Extractions

When a system fails (crashes), a kernel memory image dump (coredump) is made. This coredump can be analyzed to:

Determine why the system failed

Solve the problems that caused the system failure

After a coredump has been generated, the system administrator must use the **savecore** command to give the coredump a unique name. The coredump is then placed where it will not be overwritten in the case of another failure. This enables the system administrator to analyze the dump to determine the cause of the system failure at some later time.

The AIX/370 dump areas contain only enough space for a small number of simultaneous coredumps. When a system failure occurs, the coredump is taken to the next available coredump area. In most circumstances, this prevents previous coredumps from being overwritten should many system failures occur between the first failure and the next time the system is successfully brought up.

In the unlikely event that the dump area completely fills before the next successful boot, subsequent coredumps are not saved until the coredump area is cleared. This ensures that the earliest dumps are retained. When the system reboots after generating a coredump, existing coredumps must be moved from the dump area to another file system area for analysis.

The AIX PS/2. system does not provide the multiple dump storage mechanism of the AIX/370. Kernel coredumps are simply dumped to the specified minidisk or diskettes.

12.0 Chapter 12. Diagnosis Planning

Subtopics

12.1 Contents

12.2 About This Chapter

Since problems may occur in any computer installation, overall planning for AIX/370 must include problem diagnosis techniques. This chapter outlines the instructions that may be prepared for system operators. The responsibilities for handling various kinds of problems are explained and tools for problem diagnosis are presented.

12.3 Resolving Problems

In planning for problem diagnosis, work out instructions for the operator when problems occur. Usually, the instructions would include:

Reporting Who to contact if a problem occurs?

Information collection How, when, and which type of dump is to be performed (unless dumps are generated by the system automatically)?

Saving information Make sure that all additional information pertaining to the state of the operating system is also preserved (for example, console logs and any information on presented load).

Recovery How to restart the system (if restart is possible)?

For help in defining these and further instructions, refer to the *AIX/370 Diagnosis Guide*.

IBM provides support for resolving problems that may arise when using AIX. When problems occur, it is the system administrator responsibility to find out if:

The cause of the problem relates to software (known as ***problem determination***)?

Whether the cause of the problem can be traced to IBM programs or t programs that do not originate from IBM (known as ***problem source identification***)?

Problems that can be traced to IBM code should be reported to IBM. The reported problem may either be a problem already known to IBM, or it may be accepted by IBM as a new problem. If it is a known problem, IBM supplies an existing code correction (fix); if it is a new problem, IBM supplies a fix as soon as one becomes available.

Figure 12-1 in topic 12.3.1 summarizes the steps and responsibilities for handling problems that may occur while using the TCF cluster.

Subtopics

12.3.1 Documentation for Problem Diagnosis

Methods to analyze and handle problems that may relate to the TCF cluster are described in detail in the *AIX/370 Diagnosis Guide*.

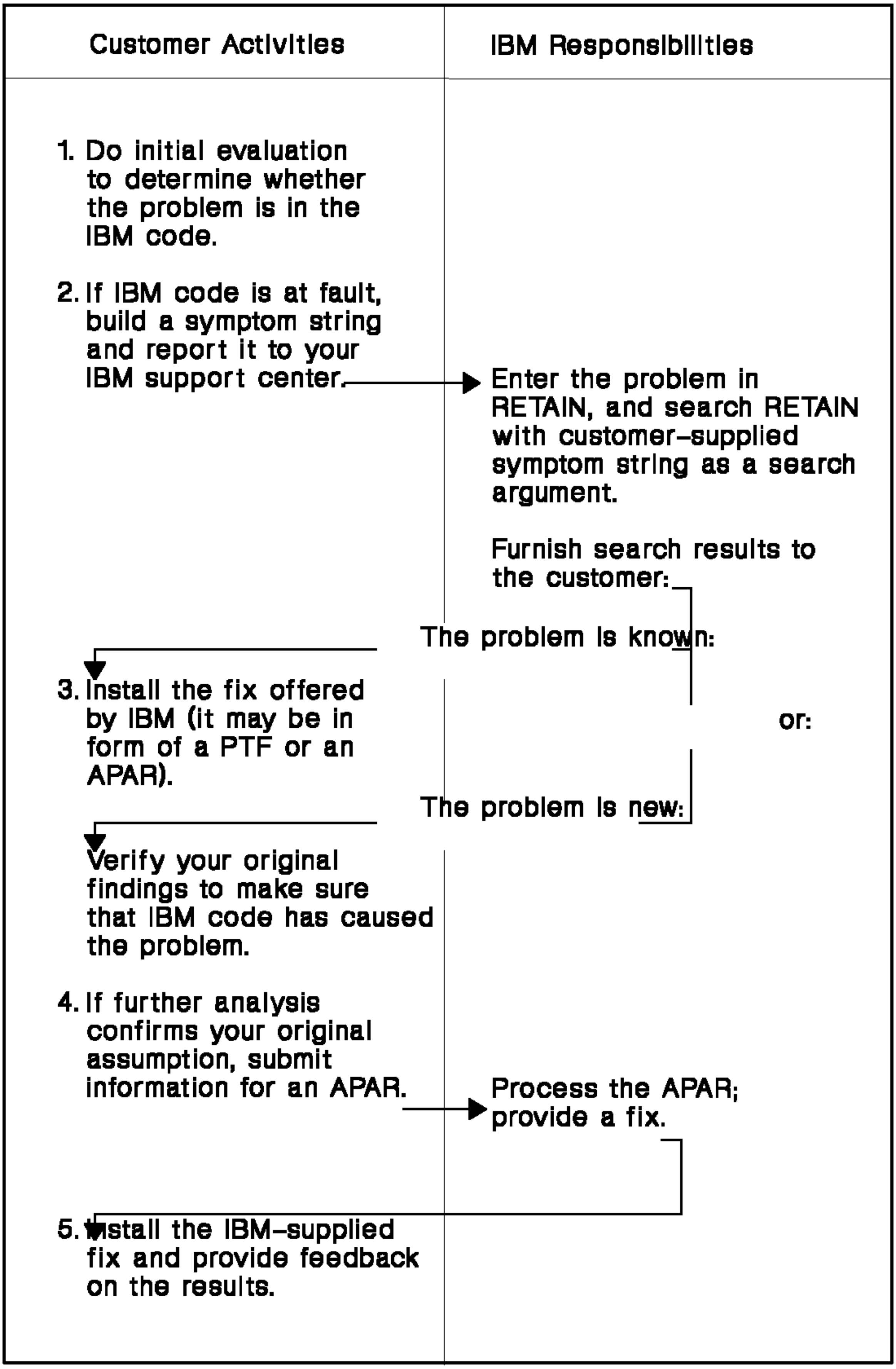


Figure 12-1. IBM Service Concept Overview

12.4 Diagnostic Aids

To isolate defective AIX code, use the following commands and utilities:

Command	Function
crash	Examines the internal structure of an operating system.
dbx	(a symbolic debugger) Can be used with C and FORTRAN programs.
dump	Dumps selected parts of an object file.
fsck	Audits and interactively repairs inconsistent conditions in file systems.
fsdb	(a file system debugger) Used to patch up a damaged file system after a system failure. Warning: This command should not be used by inexperienced personnel.
lnetstat	Provides statistics about the cluster.
netstat	Provides statistics about TCP/IP activity.
od	Views a file in various formats.
ps	Provides information about active processes in the system.
pstat	Provides information contained in certain system tables.
savecore	Provides several functions for maintaining the system dump files.
sysdump(370)	Initiates various types of dumps. This command is often used to take a ***snapshot*** of a certain section of memory in order to diagnose problems.

For information about which aid to use for a problem, refer to the *AIX Operating System Technical Reference* and *AIX Operating System Commands Reference*.

13.0 Chapter 13. Service Planning

Subtopics

13.1 Contents

13.2 About This Chapter

Part of overall planning is to decide what service is required for the system and when it should be performed. This chapter describes various types of services that are available for AIX/370. This aids in the selection of service that is most suitable for the installation.

13.3 IBM Service

Program service for AIX/370 and AIX PS/2 is available from IBM as a central service, including IBM Support Centers.

The following outlines available service and provides basic information for the user.

Subtopics

13.3.1 Preventive Service for AIX/370

Preventive service is the practice of applying all currently available fixes to the system, even if the fixes are for problems that currently do not affect the users. Preventive service is intended to prevent problems from happening. IBM provides preventive service packages to all AIX/370 customers; each package contains all Program Temporary Fixes (PTFs) made available since the last release of the product or the last preventive package, whichever is later. The preventive services package is sent on tape and can be applied to the AIX/370 system on a component-by-component basis, resulting in a cumulative upgrade of the entire system.

In addition to the AIX/370 service, the preventive service package automatically includes service to any of the AIX PS/2s that may be installed. This ensures that all installed AIX products are synchronized.

13.4 Planning for Service Application

The following section describes preventative service, history files, system backup, network-wide service and scheduling.

Subtopics

13.4.1 Preventive Service

Preventive service can install all AIX/370 problem fixes known to date. This includes fixes for problems that are not apparent at a particular installation, but may become troublesome in the future. Preventive service may entail more downtime, therefore careful resource planning for preventive service application is necessary.

13.4.2 History Files

To ensure that proper records are kept, list and retain history information of the system before and after service. History information for the system is kept in the file **/usr/lpp/**<*component*>**/lpp.hist** where *component* is the name of the software component involved. The file contains a record of when each fix was installed and to what overall level the component is currently located. The global history file **/etc/lpp/ghf** contains a summary of all the history files.

Remember to maintain equivalent service levels across all AIX products. If a certain component of the AIX/370 system is upgraded, any installed AIX PS/2 systems must have the same component upgraded to that level.

13.4.3 System Backup

Fatal errors (for example, handling errors or machine errors) can occur during a service application. Therefore, always have an entire system backup copy available to recover the system if necessary. The **updatep -r** command may be used to return to a prior service level.

The service strategy should be synchronized with the overall backup strategy for a particular installation. Ensure that enough time is allocated to perform service, as well as enough space for the backup.

13.4.4 Network-wide Service Application

When any service package is installed on AIX, it should be installed from the primary site, that is, the site where the primary copy of the root file system actually resides.

When any application is installed that involves the entire cluster, such as an incompatible kernel, the installation should be performed on all cluster sites at once. If the installation cannot be installed on all sites at the same time, refer to *Managing the AIX Operating System* for details on the special procedures needed to perform the installation.

13.4.5 Scheduling Service Application

Service procedures should be scheduled to avoid conflict with the daily system operations of the installation. Therefore,

Try to schedule service applications directly following the regula backup of the entire system.

Schedule service application for specific components. For example service can be applied to the C compiler by suspending compilations of C programs. Thus, other processes can continue without interruption.

A.0 Appendix A. Worksheets

Subtopics

A.1 Planner's Checklist

A full set of plans for the TCF cluster will require, at least, the following actions and decisions.

__ Has the hardware diagram, which shows the logical structure of the cluster (not the physical layout), been completed? Refer to Chapter 3, "The TCF Cluster" for more information.

__ Has the hardware/software diagram been completed? Refer to Chapter 3, "The TCF Cluster" for more information.

__ Has the name/address diagram been completed? Refer to Chapter 3, "The TCF Cluster" for more information.

- __ - Has the VM mode (V=V, V=R, V=F) been decided for each System/370 machine? Refer to Chapter 7, "The AIX/370 Guest System Virtual Machine" for more information.
- __ - Has the LAN type (Ethernet or Token Ring) been decided for each LAN?
- __ - Has the Internet address been decided for each site in the cluster? Refer to the *AIX Operating System TCP/IP User's Guide*.
- __ - Has a particular System/370 machine been chosen to run as the AIX/370 primary site?
- __ - Has the number and size of the virtual machines running on each System/370 been determined?
- __ - Has the software which is to be run on each virtual machine been selected?
- __ - Have names been chosen for each site in the cluster?
- __ - Have numbers been chosen for each site in the cluster?

__ Is the final cluster diagram complete? Refer to Chapter 3, "The TCF Cluster" for more information.

- __ - Has a System/370 site been selected to hold a backbone copy of the file system?
- __ - Has a site(s) been selected to hold secondary copies of the file system?
- __ - Have the hosts, which will support user terminals, been chosen?
- __ - What style of required adapter (Ethernet, Token Ring, and/or serial) has been selected for each PS/2 host?

__ Is minidisk allocation complete?

- __ - Has it been decided whether to use VM minidisks, AIX minidisks, or dedicated DASD?
- __ - Has the AIX minidisk worksheet been completed? Refer to Chapter 6, "Installation Planning" for more information.
- __ - Has the VM minidisk worksheet been completed? Refer to Chapter 6, "Installation Planning" for more information.

__ Has the backup process been planned for?

- __ - Has it been determined which sites will hold which file systems?
- __ - Has it been determined which file systems will require backup? At what intervals will backups be required?

A.2 AIX Minidisk Worksheet

Table A-1. AIX Minidisk Calculation Worksheet

Program Name or ***User File System***	**Number of 1024 byte blocks in each AIX Minidisk**					
	/ (root)	**/aix370**	**/aix370**	**tmdump**	**page**	*user*
Sub-totals						
Future Uses						
Totals						

A.3 VM Minidisk Worksheet

Table A-2. VM minidisk requirements for the AIX/370 guest system

Contents of VM minidisk	**Virtual address**	**# of FB-512 blocks or # of cylinders**	**Starting block/cylinder (or entire disk)**	**Volume I or real address**
'A' (191) Disk	191	7,000 blocks 5 cylinders		
I/M System¦	340 (FBA) 390 (CKD)			
Base System	341 (FBA) 391 (CKD)			
AIX Minidisk[2], ¦	342 (FBA) 392 (CKD)			
AIX Minidisk[2], ¦	343 (FBA) 393 (CKD)			
AIX Minidisk[2], ¦	344 (FBA) 394 (CKD)			
AIX Minidisk[2], ¦	345 (FBA) 395 (CKD)			
AIX Minidisk[2]				
AIX Minidisk[2]				

() () ()

() ¦The I/M System should be installed on a separate physical volume from the Base System volume.

() [2]These VM minidisks (virtual addresses 342 to 345 and 392 to 395) can be used to store the system-defined AIX minidisks and user AIX minidisks. If these AIX minidisks virtual addresses are not defined in the AIX/370 guest system virtual machine, the system-defined AIX minidisks and user AIX minidisks (*user*) will be installed as part of the Base System VM minidisk.

() ¦Depending on the requirements of the installation, all four of these VM minidisks may not be needed. However, if additional storage space is required on other than these four virtual addresses, the AIX/370 kernel must be regenerated. Additional virtual addresses are defined by using the AIX/370 **devices** command.

B.0 Appendix B. Examples and Completed Sample Worksheets

Subtopics

B.1 Examples and Completed Worksheets

The following section includes example diagrams and completed worksheets. They may be used as an aid during the planning process. These completed diagrams and worksheets represent the type of information that must be planned for prior to the installation of AIX/370.

Subtopics

B.1.1 Hardware Diagram Example

Figure B-1 is an example of a hardware diagram. This sketch shows the logical configuration of the TCF cluster devices. The physical layout may be entirely different.

Note: This artwork cannot be shown in softcopy.

Figure B-1. An example of a hardware diagram for the TCF cluster

B.1.2 Hardware/Software Diagram Example

Figure B-2 shows the software packages that are to be executed on each TCF cluster site.

Note: This artwork cannot be shown in softcopy.

Figure B-2. An example of a hardware/software diagram for the TCF cluster

B.1.3 Sitename/Address Diagram Example

Figure B-3 shows:

- The site name of each cluster sit
- The Internet address for each cluster sit
- The type of LAN to be use
- The virtual machines executing in each processo
- The software that executes in each virtual machine

Note: This artwork cannot be shown in softcopy.

Figure B-3. An example of a name/address diagram for the TCF cluster

B.1.4 Final TCF Cluster Diagram Example

Figure B-4 shows the final TCF cluster diagram.

Note: This artwork cannot be shown in softcopy.

```
¦                                                                         ¦
¦                                                                         ¦
¦                                                                         ¦
+-------------------------------------------------------------------------+
```

Figure B-4. An example of a final TCF cluster diagram

B.1.5 Completed Planner's Checklist

Figure B-5 is an example of a completed Planner's Checklist.

Note: This artwork cannot be shown in softcopy.

```
¦                                                                        ¦
¦                                                                        ¦
¦                                                                        ¦
+------------------------------------------------------------------------+
```

Figure B-5. An example of a completed Planner's Checklist

B.1.6 Completed AIX Minidisk Worksheet

Figure B-6 is an example of a completed AIX Minidisk Worksheet.

Note: This artwork cannot be shown in softcopy.

```
|                                                                      |
|                                                                      |
|                                                                      |
+----------------------------------------------------------------------+
```

Figure B-6. An example of a completed AIX Minidisk Worksheet

B.1.7 Completed VM Minidisk Worksheet

Figure B-7 is an example of a completed VM Minidisk Worksheet.

Note: This artwork cannot be shown in softcopy.

```
¦                                                                    ¦
¦                                                                    ¦
¦                                                                    ¦
+--------------------------------------------------------------------+
```

Figure B-7. An example of a completed VM Minidisk Worksheet

C.0 Appendix C. AIX/370 Default Device Addresses

Subtopics

C.1 Device Addresses

The following tables contain the default device addresses which are known to the AIX/370 kernel. To change or add more addresses, use the **devices** command. For more information on the **devices** command, refer to *Installing and Customizing the AIX/370 Operating System*.

Subtopics

C.1.1 DASD Addresses

Table C-1 list the default addresses for VM DASD devices.

Table C-1. Default DASD device addresses

FBA Address	CKD Address	Contents
340	390	I/M System
341	391	Base System volume
342 to 345	392 to 395	AIX minidisks

C.1.2 LAN Addresses

Table C-2 list the default addresses for LAN devices.

Table C-2. Default LAN device addresses

Ethernet	**Token-Ring**	**8232**
100 to 103	120 to 123	160 and 161

GLOSSARY Glossary

access. To obtain computing services.

access permission. A group of designations that determine who can access a particular AIX file and how the user may access the file.

account. The login directory and other information that give a user access to the system.

activity manager. A collection of system programs allowing users to manage their activities. Provides the ability to list current activities (Activity List) and to begin, cancel, hide, and activate activities.

All Points Addressable (APA) display. A display that allows each pixel to be individually addressed. An APA display allows for images to be displayed that are not made up of images predefined in character boxes. Contrast with **character display**.

allocate. To assign a resource, such as a disk file or a diskette file, to perform a specific task.

alphabetic. Pertaining to a set of letters A through Z.

alphanumeric character. Consisting of letters, numbers, and often other symbols, such as punctuation marks and mathematical symbols.

American National Standard Code for Information Interchange (ASCII). The code developed by ANSI for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII character set consists of 7-bit control characters and symbolic characters.

American National Standards Institute (ANSI). An organization sponsored by the Computer and Business Equipment Manufacturers Association for establishing voluntary industry standards.

APAR. Authorized Program Analysis Report.

application. A program or group of programs that directly apply to a particular user problem, such as inventory control, word processing, or accounts receivable.

application program. A program used to perform an application or part of an application.

argument. Numbers, letters, or words that affect the way a command works.

ASCII. See **American National Standard Code for Information Interchange**.

assembler. A computer program that converts assembly language instructions into object code.

asynchronous transmission. In data communication, a method of transmission in which the bits included in a character or block of characters occur during a specific time interval. However, the start of each character or block of characters can occur at any time during this interval. Contrast with **synchronous transmission**.

attribute. A characteristic. For example, the attribute for a displayed field could be blinking.

authorize. To grant to a user the right to communicate with, or make use of, a computer system or display station.

authorized program analysis report (APAR). A report of a problem caused by a suspected defect in a current unaltered release of a program.

auto carrier return. The system function that places carrier returns automatically within the text and on the display. This is accomplished by moving whole words that exceed the line end zone to the next line.

backend. The program that sends output to a particular device. There are two types of backends: friendly and unfriendly.

background process. (1) A process that does not require operator intervention that can be run by the computer while the work station is used to do other work. (2) A mode of program execution in which the shell does not wait for program completion before prompting the user for another command.

backup copy. A copy, usually of a file or group of files, that is kept in case the original file or files are unintentionally changed or destroyed.

backup diskette. A diskette containing information copied from a fixed disk or from another diskette. It is used in case the original information becomes unusable.

bad block. A portion of a disk that can never be used reliably.

BAL. Basic Assembler Language.

base address. The beginning address for resolving symbolic references to locations in storage.

base name. The last element to the right of a full path name. A filename specified without its parent directories.

batch printing. Queueing one or more documents to print as a separate job. The operator can type or revise additional documents at the same time. This is a background process.

batch processing. A processing method in which a program or programs process records with little or no operator action. This is a background process. Contrast with **interactive processing**.

binary. (1) Pertaining to a system of numbers to the base two; the binary digits are 0 and 1. (2) Involving a choice of two conditions, such as on-off or yes-no.

bit. Either of the binary digits 0 or 1 used in computers to store information. Eight bits make a byte. See also **byte**.

block. (1) A group of records that is recorded or processed as a unit. Same as **physical record**. (2) In data communication, a group of records that is recorded, processed, or sent as a unit. (3) A physical block in AIX is 4096 bytes long. (4) A logical block in AIX/370 and AIX PS/2 is 1024 bytes. (5) A logical block in AIX/RT is 512 bytes.

block file. A file listing the usage of blocks on a disk.

block special file. A special file that provides access to an input or output device capable of supporting a file system. See also **character special file**.

boot. To prepare a computer system for operation by loading an operating system.

bootstrap. A small program that loads larger programs during system initialization. Sometimes referred to as IPL (Initial Program Load).

Bourne shell. A flexible command language that can be customized to specific applications or user needs.

bpi. bits per inch.

bps. bits per second.

branch. In a computer program an instruction that selects one of two or more alternative sets of instructions. A conditional branch occurs only when a specified condition is met.

breakpoint. A place in a computer program, usually specified by an instruction, where execution may be interrupted by external intervention or by a monitor program.

BSD. Berkeley Software Distribution.

buffer. (1) A temporary storage unit, especially one that accepts information at one rate and delivers it at another rate. (2) An area of storage, temporarily reserved for performing input or output, into which data is read, or from which data is written.

burst pages. On continuous-form paper, pages of output that can be separated at the perforations.

byte. The amount of storage required to represent one character; a byte is 8 bits.

call. To activate a program or procedure at its entry point. Compare with **load**.

callouts. An AIX kernel parameter establishing the maximum number of scheduled activities that can be pending simultaneously.

cancel. To end a task before it is completed.

carrier return. (1) In text data, the action causing line ending formatting to be performed at the current cursor location followed by a line advance of the cursor. Equivalent to the carriage return of a typewriter. (2) A keystroke generally indicating the end of a command line.

case sensitive. Able to distinguish between uppercase and lowercase letters.

CCW. Channel Command word.

central processing unit. The part of a computer that includes the circuits that control the interpretation and execution of instructions.

character. A letter, digit, or other symbol.

character display. A display that uses a character generator to display predefined character boxes of images (characters) on the screen. This kind of display cannot address the screen any less than one character box at a time. Contrast with **All Points Addressable display**.

character key. A keyboard key that allows the user to enter the character shown on the key. Compare with **function keys**.

character position. On a display, each location that a character or symbol can occupy.

character set. A group of characters used for a specific reason; for example, the set of characters a printer can print or a keyboard can support.

character special file. A special file that provides access to an input or output device. The character interface is used for devices that do not use block I/O. See also **block special file**.

character string. A sequence of consecutive characters.

character variable. The name of a character data item whose value may be assigned or changed while the program is running.

child. (1) Pertaining to a secured resource, either a file or library, that uses the user list of a parent resource. A child resource can have only one parent resource. (2) In the AIX Operating System child is a **process** spawned by a parent process that shares the attributes of the parent process. Contrast with **parent**.

C language. A general-purpose programming language that is the primary language of the AIX Operating System.

class. Pertaining to the I/O characteristics of a device. AIX devices are classified as block or character.

close. To end an activity and remove that window from the display.

cluster. (1) Any configuration of workstations for the purpose of sharing resources (for example, Local Area Networks (LANs) and host attached workstations). (2) A group of storage locations allocated at one time.

CMS. Conversational Monitoring System.

code. (1) Instructions for the computer. (2) To write instructions for the computer; to **program**. (3) A representation of a condition, such as an error code.

code segment. See **segment**.

collating sequence. The sequence in which characters are ordered within the computer for sorting, combining, or comparing.

color display. A display device capable of displaying more than two colors and the shades produced via the two colors, as opposed to a monochrome display.

column. A vertical arrangement of text or numbers.

column headings. Text appearing near the top of columns of data for the purpose of identifying or titling.

command. A request to perform an operation or run a program. When parameters, arguments, flags, or other operands are associated with a command, the resulting character string is a single command.

command interpreter. A program (such as the Bourne or C shell) that sends instructions from the command line to the kernel.

command line. The area of the screen where commands are displayed as they are typed.

command line editing keys. Keys for editing the command line.

command name. (1) The first or principal term in a command. A command name does not include parameters, arguments, flags, or other operands. (2) The full name of a command when an abbreviated form is recognized by the computer (for example, print working directory for **pwd**).

command programming language. Facility that allows programming by the combination of commands rather than by writing statements in a conventional programming language. See **shell procedure**.

communication adapter. A hardware feature enabling a computer or device to become part of a data communication network.

compile. (1) To translate a program written in a high-level programming language into a machine language program. (2) The computer actions required to transform a source file into an executable object file.

compress. (1) To move files and libraries together on disk to create one continuous area of unused space. (2) In data communication, to delete a series of duplicate characters in a character string.

concatenate. (1) To link together. (2) To join two character strings.

concentrator. (1) In data transmission, a functional unit that permits a common medium to serve more data sources than there are channels currently available within the transmission medium. (2) Any device that combines incoming messages into a single message (concentration) or extracts individual messages from the data sent in a single transmission sequence (deconcentration).

condition. An expression in a program or procedure that can be evaluated to a value of either true or false when the program or procedure is running.

configuration. The group of machines, devices, and programs that make up a computer system. See also **system customization**.

configuration file. A file that specifies the characteristics of a system or subsystem, for example, the AIX queueing system.

consistent. Pertaining to a file system, without internal discrepancies.

console. (1) The main AIX display station. (2) A device name associated with the main AIX display station.

constant. A data item with a value that does not change. Contrast with **variable**.

context search. A search through a file for a character string.

control block. A storage area used by a program to hold control information.

control commands. Commands that allow conditional or looping logic flow in shell procedures.

control program. Part of the AIX Operating System that determines the order in which basic functions should be performed.

controlled cancel. The system action that ends the job step being run, and saves any new data already created. The job that is running can continue with the next job step.

copy. The action by which the user makes a whole or partial duplicate of already existing data.

coredump. A kernel memory image dump that is given a unique name so that it will not be over-written in case of another failure. This enables the system administrator to analyze the dump and determine the cause of failure at some later time.

coupler. A device connecting a modem to a telephone network.

CPU. Central Processing Unit.

crash. An unexpected interruption of computer service, usually due to a serious hardware or software malfunction.

CTC. Channel-to-Channel.

current directory. The directory that is the starting point for relative pathnames.

current line. The line on which the cursor is located.

current synchronization site (CSS). The site in the cluster containing the primary copy of the replicated file system.

current working directory. See **current directory**.

cursor. (1) A movable symbol (such as an underline) on a display, used to indicate to the operator where the next typed character will be placed or where the next action will be directed. (2) A marker that indicates the current data access location within a file.

cursor movement keys. The directional keys used to move the cursor.

customize. To describe (to the system) the devices, programs, users, and user defaults for a particular data processing system.

cylinder. All fixed disk or diskette tracks that can be read or written without moving the disk drive or diskette drive read/write mechanism.

daemon. See **daemon process**.

daemon process. A process begun by the root or the root shell that can be stopped only by the root. Daemon processes generally provide services that must be available at all times to more than one task or user, such as sending data to a printer.

DASD. Direct Access Storage Device.

data block. See **block**.

data communication. The transmission of data between computers, and/or remote devices (usually over long distance).

data link. The equipment and rules (protocols) used for sending and receiving data.

data stream. All information (data and control information) transmitted over a data link.

debug. (1) To detect, locate, and correct mistakes in a program. (2) To find the cause of problems detected in software.

default. A value that is used when no alternative is specified by the operator.

default directory. The directory name supplied by the operating system if none is specified.

default drive. The drive name supplied by the operating system if none is specified.

default value. A value stored in the system that is used when no other value is specified.

delete. To remove. For example, to delete a file.

dependent work station. A work station having little or no standalone capability, that must be connected to a host or server in order to provide any meaningful capability to the user.

device. An electrical or electronic machine that is designed for a specific purpose and that attaches to your computer, for example, a printer, plotter, or disk drive.

device driver. A program that operates a specific device, such as a printer, disk drive, or display.

device manager. Collection of routines that act as an intermediary between device drivers and virtual machines for complex interfaces. For example, supervisor calls from a virtual machine are examined by a device manager and are routed to the appropriate subordinate device drivers.

device name. A name reserved by the system that refers to a specific device.

diagnostic. Pertaining to the detection and isolation of an error.

diagnostic aid. A tool (procedure, program, reference manual) used to detect and isolate a device or program malfunction or error.

diagnostic routine. A computer program that recognizes, locates, and explains either a fault in equipment or a mistake in a computer program.

digit. Any of the numerals from 0 through 9.

directory. A type of file containing the names and controlling information for other files or other directories.

disable. To make nonfunctional.

discipline. Pertaining to the order in which requests are serviced, for example, first-come-first-served (fcfs) or shortest job next (sjn).

disk I/O. Fixed-disk input and output.

diskette. A thin, flexible magnetic plate that is permanently sealed in a protective cover. It can be used to store information copied from the disk or another diskette.

diskette drive. The mechanism used to read and write information on diskettes.

display device. An output unit that gives a visual representation of data.

display screen. The part of the display device that displays information visually.

display station. A device that includes a keyboard from which an operator can send information to the system and a display screen on which an operator can see the information sent to or received from the computer.

distributed file system. A file system whose files, directories, and other components are stored on different sites in a particular cluster.

distributed operating system. An operating system where multiple machines cooperate to seem like one machine.

distributed processing. Results when a user involves multiple cluster sites in a single operation--for example, by editing a remote file and starting a task on another cluster site using the **on**, **fast**, **fastsite**, and **migrate** commands.

Distributed Services (DS). A licensed program that allows you to share files with other AIX systems in a network. You can mount the file systems located on other AIX systems to created file trees that are independent of the file systems.

DOS. Disk Operating System.

dumb terminal. See **fixed-function terminal**

dumb workstation. See **fixed-function workstation**

dump. (1) To copy the contents of all or part of storage, usually to an output device. (2) Data that has been dumped.

dump diskette. A diskette that contains a dump or is prepared to receive a dump.

dump formatter. Program for analyzing a dump.

EBCDIC. See **extended binary-coded decimal interchange code.**

EBCDIC character. Any one of the symbols included in the 8-bit EBCDIC set.

edit. To modify the form or format of data.

edit buffer. A temporary storage area used by an editor.

editor. A program used to enter and modify programs, text, and other types of documents and data.

emulation. Imitation; for example, when one computer imitates the characteristics of another computer.

enable. To make functional.

enter. To send information to the computer by pressing the **Enter** key.

entry. A single input operation on a work station.

environment. The settings for shell variables and paths associated with each process. These variables can be modified later by the user.

EREP. Environmental Recording Edit and Print program.

error-correct backspace. An editing key that performs editing based on a cursor position; the cursor is moved one position toward the beginning of the line, the character at the new cursor location is deleted, and all characters following the cursor are moved one position toward the beginning of the line (to fill the vacancy left by the deleted element).

escape character. A character that suppresses the special meaning of one or more characters that follow.

ESSL. Engineering and Scientific Subroutine Library.

Ethernet. A physical medium through which computers in the same or different clusters can communicate and share files.

exit value. A numeric value that a command returns to indicate whether it completed successfully. Some commands return exit values that give other information, such as whether a file exists. Shell programs can test exit values to control branching and looping. Exit values are also called Return Codes.

expression. A representation of a value. For example, variables and constants appearing alone or in combination with operators.

extended binary-coded decimal interchange code (EBCDIC). A set of 256 eight-bit characters.

feature. A programming or hardware option, usually available at an extra cost.

field. (1) An area in a record or panel used to contain a particular category of data. (2) The smallest component of a record that can be referred to by a name.

FIFO. See **first-in-first-out**.

file. A collection of related data that is stored and retrieved by an assigned name.

file name. The name used by a program to identify a file. See also

label.

filename. In DOS, that portion of the file name that precedes the extension.

file specification (filespec). The name and location of a file. In DOS a file specification consists of a drive specifier, a path name, and a file name.

file system. A collection of files and directories stored on logical and physical devices (such as disks) and logically organized in a hierarchical fashion.

filetab. An AIX kernel parameter establishing the maximum number of files that can be open simultaneously.

filter. A command that reads standard input data, modifies the data, and sends it to standard output.

first-in-first-out (FIFO). A named permanent pipe. A FIFO allows two unrelated processes to exchange information using a pipe connection.

fixed disk. A flat, circular, non-removable plate with a magnetized surface layer on which data can be stored by magnetic recording.

fixed-disk drive. The mechanism used to read and write information on fixed disk.

fixed-function terminal. A terminal designed to perform a particular set of operations and that cannot be programmed to perform other functions. Contrast with programmable terminal.

fixed-function workstation. A workstation that is designed to perform a particular set of operations and that cannot be programmed to perform other functions. Contrast with programmable workstation.

flag. A modifier that appears on a command line with the command name that defines the action of the command. Flags in the AIX Operating System are almost always preceded by a dash.

font. A family or assortment of characters of a given size and style.

foreground. A mode of program execution in which the shell waits for the program specified on the command line to complete before returning your prompt.

format. (1) A defined arrangement of such things as characters, fields, and lines, usually used for displays, printouts, or files. (2) The pattern which determines how data is recorded.

formatted diskette. A diskette on which control information for a particular computer system has been written but which may or may not contain any data.

FORTRAN. A programming language primarily used to express computer programs by arithmetic formulas and numeric computations.

free list. A list of available space on each file system. This is sometimes called the free-block list.

free-block list. See **free list**.

full path name. The name of any directory or file expressed as a string of directories and files beginning with the root directory.

function. A synonym for procedure. The C language treats a function as a data type that contains executable code and returns a single value to the calling routine.

function keys. Keys that request actions but do not display or print characters. Included are the keys that normally produce a printed character, but when used with the code key produce a function instead. Compare with **character key**.

gateway. A functional unit that connects two computer networks.

generation. For some remote systems, the translation of configuration information into machine language.

Gid. See **group number**.

global. Pertains to information available to more than one program or subroutine.

global action. An action having general applicability, independent of the context established by any task.

global character. The special characters * and ? that can be used in a file specification to match one or more characters. For example, placing a ? in a file specification means any character can be in that position. See **pattern-matching character**.

global search. The process of having the system look through a document for specific characters, words, or groups of characters.

global variable. A symbol defined in one program module but used in other independently assembled program modules.

graphic character. A character that can be displayed or printed.

group name. A name that uniquely identifies a group of users to the system.

group number (Gid). A unique number assigned to a group of related users. The group number can often be substituted in commands that take a group name as an argument.

hardware. The equipment, as opposed to the programming, of a computer system.

header. Constant text that is formatted to be in the top margin of one or more pages.

header label. A special set of records on a diskette describing the contents of the diskette.

here document. Data contained within a shell program or procedure (also called **inline input**).

hexadecimal. Pertaining to a system of numbers using base sixteen; hexadecimal digits range from 0 (zero) through 9 (nine) and A (ten) through F (fifteen).

hierarchical tree structure. The organization of files on AIX, similar to tree-structured directories, with each file like a small branch of a larger branch that represents the file's parent directory. A directory can also be contained in another higher level directory, with the parent of all directories represented by the tree's root (***root*** or ***root directory***).

highlight. To emphasize an area on the display by any of several methods, such as brightening the area or reversing the color of characters within the area.

history. A C-shell mechanism that lists previously executed commands. These commands can be re-executed with the **!** command.

history file. A file containing a log of system actions and operator responses.

hog factor. In system accounting, an analysis of how many times each command was run, how much processor time and memory it used, and how intensive that use was.

home directory. The directory a user accesses when he logs in. Here he may create and delete files and directories to organize his work. Synonym for **login directory**.

home site. The computer that stores the modifiable copy of a user's home directory. This is the cluster site with the primary copy of his home directory if it is replicated. A user typically logs in to the computer that is his home site.

I/O. See **input/output**.

ID. Identification.

IDAW. Indirect Addressing Word.

IEEE. Institute of Electrical and Electronics Engineers.

IF expressions. Expressions within a procedure, used to test for a condition.

indirect block. A block containing pointers to other blocks. Indirect blocks can be single-indirect, double-indirect, or triple-indirect.

INed. A full screen editor that also features windows.

informational message. A message providing information to the operator, that does not require a response.

initial program load (IPL). The process of loading the system programs and preparing the system to run jobs. See **initialize, bootstrap**.

initialize. To set counters, switches, addresses, or contents of storage to zero or other starting values at the beginning of, or at prescribed points in, the operation of a computer routine.

inline input. See **here document**.

inode. The internal structure for managing files in the system. Inodes contain all of the information pertaining to the node, type, owner, and location of a file. A table of inodes is stored near the beginning of a file system.

i-number. A number specifying a particular inode on a file system.

inodetab. An AIX kernel parameter that establishes a table in memory for storing copies of inodes for all active files.

input. Data to be processed.

input device. Physical devices used to provide data to a computer.

input file. A file opened by a program so that the program can read from that file.

input list. A list of variables to which values are assigned from input data.

input redirection. The specification of an input source other than the standard one.

input-output file. A file opened for input and output use.

input-output device number. A value assigned to a device driver by the guest operating system or to the virtual device by the virtual resource manager. This number uniquely identifies the device regardless of whether it is real or virtual.

input/output (I/O). Pertaining to either input, output, or both between a computer and a device.

intelligent terminal. See **programmable terminal**

intelligent workstation. See **programmable workstation**

interactive processing. A processing method in which each system user action causes response from the program or the system. Contrast with **batch processing**.

interface. A shared boundary between two or more entities. An interface might be a hardware component to link two devices together or it might be a portion of storage or registers accessed by two or more computer programs.

interleave factor. Specification of the ratio between contiguous physical blocks (on a fixed-disk) and logically contiguous blocks (as in a file).

interrupt. (1) To temporarily stop a process. (2) In data communication, to take an action at a receiving station that causes the sending station to end a transmission. (3) A signal sent by an I/O device to the processor when an error has occurred or when assistance is needed to complete I/O. An interrupt usually suspends execution of the currently executing program.

interrupt character. A key sequence (**Alt-Pause** on some systems) typed in to cancel a foreground process.

IPL. Initial Program Load.

job. (1) A unit of work to be done by a system. (2) One or more related procedures or programs grouped into a procedure.

job control. A feature that lets the system accept your commands to stop and start processes (jobs) and move them between background and foreground. The commands **ps** and **jobs** report the status of jobs, (each of which is assigned a Process Identification Number or PID to show its process status), and the **kill** command can be used to stop them.

job number. A number assigned to a background process when it is started. The job number is displayed when the process is started and when the **jobs** or **ps** command is invoked. It can also be used to kill the process.

job queue. A list, on disk, of jobs waiting to be processed by the system.

justify. To print a document with even right and left margins.

kbuffers. An AIX kernel parameter establishing the number of buffers that can be used by the kernel.

K-byte (Kb). See **kilobyte**.

kernel. The memory-resident nucleus of the AIX Operating System containing functions needed immediately and frequently. The kernel supervises the input and output, manages and controls the hardware, and schedules the user processes for execution.

kernel parameters. Variables that specify how the kernel allocates certain system resources.

key pad. A physical grouping of keys on a keyboard (for example, numeric key pad, and cursor key pad).

keyboard. An input device consisting of various keys allowing the user to input data, control cursor and pointer locations, and to control the dialog between the user and the display station

keylock feature. A security feature in which a lock and key can be used to restrict the use of the display station.

keyword. One of the predefined words of a programming language; a reserved word.

keyword argument. One type of variable assignment that can be made on the command line.

kill. An AIX Operating System command that stops a process.

kill character. The character that is used to delete a line of characters entered after the user's prompt.

kilobyte. 1024 bytes.

kprocs. An AIX kernel parameter establishing the maximum number of processes that the kernel can run simultaneously.

label. (1) The name in the disk or diskette volume table of contents that identifies a file. See also **file name.** (2) The field of an instruction that assigns a symbolic name to the location at which the instruction begins, or such a symbolic name.

LAN. Local Area Network.

left margin. The area on a page between the left paper edge and the leftmost character position on the page.

left-adjust. The process of aligning lines of text at the left margin or at a tab setting such that the leftmost character in the line or file is in the leftmost position. Contrast with **right-adjust**.

library. A collection of functions, subroutines, or other data.

licensed program (LP). Software programs that remain the property of the manufacturer, for which customers pay a license fee.

line editor. An editor that modifies the contents of a file one line at a time.

linefeed. An ASCII character that causes an output device to move forward one line.

link. A connection between an inode and one or more file names associated with it. Synonym for **UNIX link** or **hard link**.

literal. A symbol or a quantity in a source program that is itself data, rather than a reference to data.

load. (1) To move data or programs into storage. (2) To place a diskette into a diskette drive, or a magazine into a diskette magazine drive. (3) To insert paper into a printer.

loader. A program that reads run files into main storage, thus preparing them for execution.

local. Pertaining to a device directly connected to your system without the use of a communication line. Contrast with **remote**.

<LOCAL> alias. The <LOCAL> alias can translate into different strings on different cluster sites for different processes. When <LOCAL> is the first component of the destination name for a symbolic link, it is replaced with its alias string, normally /**machinename**.

local area network (LAN). A physical medium that allows computers in the same or different clusters to communicate and share files. Ethernet and Token-Ring are two examples of a LAN.

local cluster site. The site on a cluster that the user is logged in to. The term **local** normally refers to a TCF cluster site.

<LOCAL> file system. The part of the root file system hierarchy comprising system directories and files (such as the /etc/motd "message of the day" file) defined uniquely on a particular computer in the cluster. These files are not replicated. The name of the <LOCAL> file system appears in response to the **site-l** command.

location transparency. Allows an object to change location without the user's or program's knowledge if that location is not part of the object's name. For example, **/u/joe/glossary** may have been a file on **eyore** last week, but it is a file on **pooh** this week. Joe does not need to know that the file was on either **eyore** or **pooh**. If, however, Joe wants to find out where the site is located, he may invoke the **where** command.

log. To record; for example, to log all messages on the system printer. A list of this type is called a log, such as an error log.

log in. To begin a session at a display station.

log off. See **log out**.

log on. See **log in**.

log out. To end a session at a display station.

logical device. A file for conducting input or output with a physical device.

login directory. See **home directory**.

login shell. The program, or command interpreter, started for a user at log in.

login user ID. The ID the user uses to log in. The system uses this ID to trace all user actions to their source.

loop. A sequence of instructions performed repeatedly until an ending condition is reached.

LP. See **licensed program**.

macro. A set of statements defining the name of, format of, and conditions for generating a sequence of assembler statements from a single source statement.

mailbox. An area designated for storage of mail messages directed to a specific system user.

main storage. The part of the processing unit where programs are run.

maintenance system. A special version of the AIX Operating System which is loaded from diskette and used to perform system management tasks.

major device number. A system identification number for each device or type of device.

mapped files. Files on the fixed-disk that are accessed as if they are in memory.

mask. A pattern of characters that controls the keeping, deleting, or testing of portions of another pattern of characters.

matrix. An array arranged in rows and columns.

maxprocs. An AIX kernel parameter establishing the maximum number of processes that can be run simultaneously by a user.

MBCS. multibyte character set

M-byte (Mb). Megabyte (1,048,576 bytes).

memory. Storage on electronic chips. Examples of memory are random access memory, read only memory, or registers. See **storage**.

menu. A displayed list of items from which an operator can make a selection.

message. (1) A response from the system to inform the operator of a condition which may affect further processing of a current program. (2) Information sent from one user in a multi-user operating system to another.

minidisk. A logical division of a fixed disk.

minor device number. A number used to specify various types of information about a particular device, for example, to distinguish among several printers of the same type.

mode word. An inode field that describes the type and state of the inode.

modem. See **modulator-demodulator**.

modulation. Changing the frequency or size of one signal by using the frequency or size of another signal.

modulator-demodulator (modem). A device that converts data from the computer to a signal that can be transmitted on a communication line, and converts the signal received to data for the computer.

module. (1) A discrete programming unit that usually performs a specific task or set of tasks. Modules are subroutines and calling programs that are assembled separately, then linked to make a complete program. (2) See **load module**.

mount. To make a file system accessible.

mount point. Any directory which has a file system mounted to it.

mountab. An AIX kernel parameter establishing the maximum number of file systems that can be mounted simultaneously.

multibyte character set (MBCS). A method of encoding a set of written symbols so that the coded set is large enough to encompass all of the world's written languages. The MBCS system handles character sets which range in encoding size from one to four bytes.

multiprogramming. The processing of two or more programs at the same time on the same logical system.

multivolume file. A diskette file occupying more than one diskette.

multi-user environment. A computer system that provides terminals and keyboards for more than one user at the same time.

MVS. Multiple Virtual Storage.

nest. To incorporate a structure or structures of some kind into a structure of the same kind. For example, to nest one loop (the nested loop) within another loop (the nesting loop); to nest one subroutine (the nested subroutine) within another subroutine (the nesting subroutine).

network. A collection of computers that can communicate with each other. A network can consist of several interconnected computers or one computer with a number of remote terminals connected to it. Any of a variety of communication media can be used, such as RS-232, Ethernet, Token-Ring, or PC Net.

Network File System (NFS). A licensed program that allows you to share files with other computers in one or more networks that have a variety of machine types and operating systems. You can mount file systems located on network servers and use remote files as if they were on your workstations by creating file trees that are independent of the file systems.

new-line character. A control character that causes the print or display position to move to the first position on the next line.

node. An individual element of a full pathname. Nodes are separated by slashes (/).

null. Having no value, containing nothing.

null character (NUL). The character hex 00, used to represent the absence of a printed or displayed character.

numeric. Pertaining to any of the digits 0 through 9.

object code. Machine-executable instructions, usually generated by a compiler from source code written in a higher level language. It consists of directly executable machine code. For programs that must be linked, object code consists of relocatable machine code.

octal. A base eight numbering system.

OCO. Object Code Only.

online. Being controlled directly by, or communicating directly with, the computer, or both.

open. To make a file available to a program for processing.

operating system. Software that controls the running of programs; in addition, an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

operation. A specific action (such as move, add, multiply, load) that the computer performs when requested.

operator. A symbol representing an operation to be done.

output. The result of processing data.

output devices. Physical devices used by a computer to present data to a user.

output file. A file that is opened by a program so that the program can write to that file.

output redirection. The specification of an output destination other than the standard one.

overflow condition. A condition that occurs when part of the output of an operation exceeds the capacity of the intended storage unit.

override. (1) A parameter or value that replaces a previous parameter or value. (2) To replace a parameter or value.

overwrite. To write output into a storage or file space that is already occupied by data.

owner. The user who has the highest level of access authority to a data object or action, as defined by the object or action.

pad. To fill unused positions in a field with dummy data, usually zeros or blanks.

page. A block of instructions, data, or both.

page space. The area on a fixed disk that temporarily stores instructions or data currently being run. See also **minidisk**.

pagination. The process of adjusting text to fit within margins and/or page boundaries.

paging. The action of transferring instructions, data, or both between real storage and external page storage.

PANIC. An error message generated by the kernel indicating that an error has occurred which is sufficiently severe to prohibit kernel recovery.

parallel processing. The condition in which multiple tasks are being performed simultaneously within the same activity.

parameter. Information that the user supplies to a command or function.

parent. Pertaining to a secured resource, either a file or library, whose user list is shared with one or more other files or libraries. Contrast with **child**.

parent directory. The directory one level above the current directory.

partition. See **minidisk**.

Pascal. A high-level, general purpose programming language, related to ALGOL. Programs written in Pascal are block structured and consist of independent routines. They can run on different computers with little or no modification.

password. A string of characters that, when entered along with a user identification, allows an operator to login to the system.

password security. A program product option that helps prevent the unauthorized use of a display station by checking the password entered by each operator at log in.

path name. The sequential list of directory name(s) that identify the location of a particular directory, and directory name(s) and file name that identify the location of a particular file in the file hierarchy. The path name is displayed in response to the **pwd** (print working directory) command (the ~, or tilde, may appear if you're in your home directory). Each file has a full path name, beginning with / (the root directory) and ending with the file's name. The file's relative path name does not begin with /.

pattern-matching character. Special characters such as * or ? that can be used in search patterns. They are sometimes used in a file specification to match one or more characters. For example, placing a ? in a file specification means any character can be in that position. Pattern-matching characters are also called wildcards.

PC. Personal Computer.

permission code. A three-digit octal code, or a nine-letter alphabetic code, indicating the access permissions. The access permissions are read, write, and execute.

permission field. One of the three-character fields within the permissions column of a directory listing indicating the read, write, and run permissions for the file or directory owner, group, and all others.

phase. One of several stages of file system checking and repair performed by the **fsck** command.

physical device. See **device**.

physical file. An indexed file containing data for which one or more alternative indexes have been created.

physical record. (1) A group of records recorded or processed as a unit. Same as **block**. (2) A unit of data moved into or out of the computer.

PID. See **process ID**.

pipe. To direct the data so that the output from one process becomes the input to another process.

pipeline. A direct, one-way connection between two or more processes.

pitch. A unit of width of typewriter type, based on the number of times a letter can be set in a linear inch. For example, 10-pitch type has 10 characters per inch.

platen. The support mechanism for paper on a printer, commonly cylindrical, against which printing mechanisms strike to produce an impression.

pointer. (1) A logical connection between physical blocks. (2) A link to something else. (3) An address.

port. (1) To make the programming changes necessary to allow a program that runs on one type of computer to run on another type of computer. (2) A part of the system unit or remote controller to which cables for display stations and printers are attached.

position. The location of a character in a series, as in a record, a displayed message, or a computer printout.

positional parameter. A shell facility for assigning values from the command line to variables in a program.

POSIX. Portable Operating System for Computer Environments.

preprocessor. (1) A functional unit that effects preparatory computation or organization. (2) A program that examines the source program for preprocessor statements which are then executed, resulting in the alteration of the source program.

primary copy. Each replicated file system has a copy designated as the **primary copy**, which is the copy that may be modified. It resides on the **primary site** and its purpose is to guarantee that file updates are kept consistent.

primary site. The cluster site that maintains the primary copy of a replicated file system.

print queue. A file containing a list of the names of files waiting to be printed.

printout. Information from the computer produced by a printer.

priority. The relative ranking of items. For example, a job with high priority in the job queue will be run before one with medium or low priority.

priority number. A number that establishes the relative priority of printer requests.

privileged user. The account with superuser authority.

problem determination. The process of identifying why the system is not working. Often this process identifies programs, equipment, data communication facilities, or user errors as the source of the problem.

problem determination procedure. A prescribed sequence of steps aimed at recovery from, or circumvention of, problem conditions.

procedure. See **shell procedure**.

process. A program now running. A **foreground** process executes as soon as you type in the command line and completes before returning the system prompt to accept your next command. You can start one or more **background processes** to run independently while you type in a separate command for another process to run in the foreground.

process accounting. An analysis of the use each process makes of the processing unit, memory, and I/O resources.

process ID (PID). A unique number assigned to a process that is running.

process transparency. The ability to execute and control tasks on any site in the cluster, regardless of where the user is logged in (to find out where that is, type the **site** command). The same system calls and commands are used, no matter where the process is located. For example, a remote job is stopped the same way that a local job is stopped.

profile. (1) A file containing customized settings for a system or user (2) Data describing the significant features of a user, program, or device.

program. A set of instructions for the computer to interpret and execute.

programmable terminal. A user terminal that has computational capability and that can be programmed to perform user-determined functions. Contrast with fixed-function terminal.

programmable workstation. A workstation that has computational capability and that can be programmed to perform user-determined functions. Contrast with fixed-function terminal.

program temporary fix (PTF). A temporary solution or by-pass of a problem diagnosed by IBM as resulting from a defect in a current unaltered release of the program.

prompt. A displayed request for information or operator action.

propagation time. The time necessary for a signal to travel from one point on a communication line to another.

protocol. In data communication, the rules for transferring data.

protocol procedure. A process that implements a function for a device manager. For example, a virtual terminal manager may use a protocol procedure to interpret the meaning of keystrokes.

PR/SM. Process Resource/System Manager.

qdaemon. The daemon process that maintains a list of outstanding jobs and sends them to the specified device at the appropriate time.

queue. A line or list formed by items waiting to be processed.

queued message. A message from the system that is added to a list of messages stored in a file for viewing by the user at a later time. This is in contrast to a message that is sent directly to the screen for the user to see immediately.

quit. A key, command, or action that tells the system to return to a previous state or stop a process.

quote. To mask the special meaning of certain characters; to cause them to be taken literally.

radix. The positive integer by which the weight of the digit place is multiplied to obtain the weight of the digit place with the next higher weight; for example, in the decimal numeration table, the radix of each digit place is 10, in a binquinary code the radix of each fives position is 2.

random access. An access mode in which records can be read from, written to, or removed from a file in any order.

ratfor. Rational FORTRAN.

read-only. Pertaining to file system mounting, a condition that allows data to be read, but not modified.

real storage. The main storage in a virtual storage machine.

recovery procedure. (1) An action performed by the operator when an error message appears on the display screen. Usually, this action permits the program to continue or permits the operator to run the next job. (2) The method of returning the system to the point where a major system error occurred and running the recent critical jobs again.

redirect. To divert data from a process to a file or device to which it would not normally go.

reference count. In an inode, a record of the total number of directory entries that refer to the inode.

relational expression. A logical statement describing the relationship (such as greater than or equal) of two arithmetic expressions or data items.

relational operator. The reserved words or symbols used to express a relational condition or a relational expression.

relative address. An address specified relative to the address of a symbol. When a program is relocated, the addresses themselves will change, but the specification of relative addresses remains the same.

relative addressing. A means of addressing instructions and data areas by designating their locations relative to some symbol.

relative path name. The name of a directory or file expressed as a sequence of directories followed by a file name, beginning from the current directory.

remote. Pertaining to a system or device that is connected to your system through a communication line. Contrast with **local**.

remote cluster site. A site on the cluster that the user is not logged in to. The term **remote** normally refers to a TCF cluster site.

replicated root file system. The replicated root file system is a file system with key common files and directories for basic system operation. Almost all system binaries, programs and libraries are in the replicated root file system. Other user and system file systems (like the local file system) are mounted on top of directories in the replicated root file system.

reserved character. A character or symbol that has a special (non-literal) meaning unless quoted.

reserved word. A word that is defined in a programming language for a special purpose, and that must not appear as a user-declared identifier.

reset. To return a device or circuit to a clear state.

restore. To return to an original value or image. For example, to restore a library from diskette.

right adjust. The process of aligning lines of text at the right margin or tab setting such that the rightmost character in the line or file is in the rightmost position.

right justify. See right align.

right margin. The area on a page between the last text character and the right upper edge.

right-adjust. To place or move an entry in a field so that the rightmost character of the field is in the rightmost position. Contrast with **left-adjust**.

root. (1) Another name sometimes used for superuser. (2) The main file system to which others are appended.

root directory. The top level of a tree-structured directory system.

routine. A set of statements in a program causing the system to perform an operation or a series of related operations.

RSCS. Remote Spooling Communications Subsystem.

RTM. Real Time Monitor.

run. To cause a program, utility, or other machine function to be performed.

run-time environment. A collection of subroutines and shell variables that provide commonly used functions and information for system components.

SCCS. Source Code Control System.

scratch file. A file, usually used as a work file, that exists until the program that uses it ends.

screen. See **display screen**.

scroll. To move information vertically or horizontally to bring into view information that is outside the display screen boundaries.

secondary copy. A read-only copy of the primary copy of a replicated file system. Files in the secondary copy are automatically modified or deleted when the corresponding file in the primary copy is modified or deleted. New files added to the primary copy will be automatically added to the secondary copy only if the appropriate **fstore** value has been set.

sector. (1) An area on a disk track or a diskette track reserved to record information. (2) The smallest amount of information that can be written to or read from a disk or diskette during a single read or write operation.

security. The protection of data, system operations, and devices from accidental or intentional ruin, damage, or exposure.

segment. A contiguous area of virtual storage allocated to a job or system task. A program segment can be run by itself, even if the whole program is not in main storage.

semantic transparency. Allow the same command to function identically from all cluster sites. It provides, for example, for the **grep** command to have the same options and give the same results no matter where it is invoked.

semaphore. An indicator used to control access to a file: for example, in a multi-user application, a flag that prevents simultaneous access to a file.

separator. A character used to separate parts of a command or file.

sequential access. An access method in which records are read from, written to, or removed from a file based on the logical order of the records in the file.

server. A program that handles protocol, queuing, routing, and other tasks necessary for data transfer between devices in a computer system.

session. The period of time during which programs or devices can communicate with each other.

session records. In the accounting system, a record of time connected and line usage for connected display stations, produced from log in and log out records.

set flags. Flags that can be put into effect with the shell set command.

shadow page table. A table that maps real storage allocations (first-level storage) to a virtual machine's virtual storage (third-level storage) for use by the real machine in its paging options.

shared printer. A printer that is used by more than one work station.

shell. See **shell program**.

shell options. The shell provides two different types of options. **Set options** are put into effect with the **set** command and alter the way the shell runs. **Command line options** are entered on the command line (but not with the **set** command) and alter the way the shell starts.

shell procedure. A series of commands combined in a file that carry out a particular function when the file is run or when the file is specified as an argument to the **sh** command. Shell procedures are frequently called shell scripts.

shell program. A program that accepts and interprets commands for the operating system (there is an AIX shell program and a DOS shell program).

shell prompt. The character string on the command line indicating the system can accept a command (typically the **$** character).

shell script. See **shell procedure**.

shell variables. Facilities of the shell program for assigning variable values to names.

size field. In an inode, a field that indicates the size, in bytes, of the file associated with the inode.

SNOBOL. A programming language designed for string processing and pattern matching.

software. Programs.

sort. To rearrange some or all of a group of items based upon the contents or characteristics of those items.

source diskette. The diskette containing data to be copied, compared, restored, or backed up.

source program. A set of instructions written in a programming language, that must be translated to machine language and compiled before the program can be run.

special character. A character other than an alphabetic or numeric character. For example, *, +, and % are special characters.

special file. Special files are used in the AIX system to provide an interface to input/output devices. There is at least one special file for each device connected to the computer. Contrast with **directory** and **file**. See also **block special file** and **character special file**.

spool file. (1) A disk file containing output that has been saved for later printing. (2) A file used in transmitting data among devices.

SRM. System Resource Manager.

standalone shell. A limited version of the shell program used for system maintenance.

standalone work station. A work station that can be used to preform tasks independent of (without being connected to) other resources such as servers or host systems.

standard error. The place where many programs place error messages.

standard input. The primary source of data going into a command. Standard input comes from the keyboard unless redirection or piping is used, in which case standard input can be from a file or the output from another command.

standard output. The primary destination of data coming from a command. Standard output goes to the display unless redirection or piping is used, in which case standard output can be to a file or another command.

stanza. A group of lines in a file that together have a common function. Stanzas are usually separated by blank lines, and each stanza has a name.

statement. An instruction in a program or procedure.

status. (1) The current condition or state of a program or device. For example, the status of a printer. (2) The condition of the hardware or software, usually represented in a status code.

storage. (1) The location of saved information. (2) In contrast to memory, the saving of information on physical devices such as disk or tape. See **memory**.

storage device. A device for storing and/or retrieving data.

string. A series of characters to be taken literally by the system. A string may be specified for a context search, for instance, or for global substitutions.

su. (1) An AIX command that runs a shell and allows you to operate there with the privileges of the specified **user** (by default **root**). (2) See **superuser**.

subdirectory. A directory contained within another directory in the file system hierarchy.

subprogram. A program invoked by another program, such as a subshell.

subroutine. (1) A sequenced set of statements that may be used in one or more computer programs and at one or more points in a computer program. (2) A routine that can be part of another routine.

subscript. An integer or variable whose value refers to a particular element in a table or an array.

subshell. An instance of the shell program started from an existing shell program.

substitution. A procedure used by a text editor like **ed** or **vi** to replace one specified string of characters with another. If a global substitution is made, all occurrences of the specified text pattern are replaced with the new one.

substring. A part of a character string.

subsystem. A secondary or subordinate system, usually capable of operating independently of, or synchronously with, a controlling system.

superblock. The most critical part of the file system containing information about every allocation or deallocation of a block in the file system.

superuser. (1) Super user authority. (2) Root permissions.

supervisor. The part of the AIX/370 Operating System control program that coordinates the use of resources and maintains the flow of processing unit operations.

synchronous. Occurring in a regular or predictable sequence.

synchronous transmission. In data communication, a method of transmission in which the sending and receiving of characters is controlled by timing signals. Contrast with **asynchronous transmission**.

system. The computer and its associated devices and programs.

superuser. The user who can operate without the restrictions designed to prevent data loss or damage to the system (User ID 0). Also known as **superuser** or **su**.

superuser authority. The unrestricted ability to access and modify any part of the operating system associated with the user who manages the system. The authority obtained when one logs in as **root**.

symbolic link. Type of file that contains the path name to another file as a directory; it functions as a pointer to the other file or directory. See **link**.

system administrator. The person at a computer installation who designs, controls, and manages the use of the computer system.

system call. A request by an active process for a service by the system kernel.

system customization. A process of specifying the devices, programs, and users for a particular data processing system.

system date. The date assigned by the system user during setup and maintained by the system.

system dump. A copy of memory from all active programs (and their associated data) whenever an error stops the system. Contrast with **task dump**.

system management. The tasks involved in maintaining the system in good working order and modifying the system to meet changing requirements.

system parameters. See **kernel parameters**.

system primary site. The machine (cluster site) designated to hold the primary copy of the replicated root file system. When files are changed in the replicated root file system, the primary site for the cluster must be available.

system profile. A file containing the default values used in system operations.

system-replicated file system. One that contains files and directories accessed by many users regardless of the users' specific applications. These system files, programs and directories are replicated on different sites in a cluster.

system unit. The part of the system that contains the processing unit, the disk drives, and the diskette drives.

system user. A person, process, or other resource that uses the facilities of a computer system.

systems network architecture (SNA). A set of rules for controlling the transfer of information in a data communication network.

target diskette. The diskette to be used to receive data from a source diskette.

task. A basic unit of work to be performed. Examples are a user task, a server task, and a processor task.

task dump. A copy of memory from a program that failed (and its associated data). Contrast with **system dump**.

Transparent Computing Facility (TCF). A facility that automatically allows for data, process, name, location and semantic transparency. Process transparency is the ability to execute and control tasks on any cluster site, no matter where the user program is currently executing. A TCF LPP is required to obtain support.

TCF cluster. A group of computers operating under the AIX Operating System and using the Transparent Computing Facility (TCF).

TCP/IP. Transmission Control Protocol/Internet Protocol.

terminal. An input/output device containing a keyboard and either a display device or a printer. Terminals usually are connected to a computer and allow a person to interact with the computer.

text. A type of data consisting of a set of linguistic characters (for example, alphabet, numbers, and symbols) and formatting controls.

text application. A program defined for the purpose of processing text data (for example, memos, reports, and letters).

text editing program. See **editor** and **text application**.

texttab. A kernel parameter establishing the size of the text table, in memory, that contains one entry each active, shared program text segment.

Token-Ring network. A network that uses a ring topology, in which tokens are passed in the circuit from node to node. A node ready to send can capture the token and insert data for transmission.

trace. To record data that provides a history of events occurring in the system.

trace table. A storage area into which a record of the performance of computer program instructions is stored.

track. A circular path on the surface of a fixed disk, diskette, magnetic tape, or CD ROM on which information is magnetically recorded and from which recorded information is read.

transfer. To move data from one location to another in a computer system or between two or more systems.

transmission control characters. In data communication, special characters that are included in a message to control communication over a data link. For example, the sending station and the receiving station use transmission control characters to exchange information; the receiving station uses transmission control characters to indicate errors in data it receives.

transparency. The obscuring of machine boundaries in a distributed system. The AIX/370 system supports several kinds of transparency, including name, location, semantic, data, and process transparency.

trap. An unprogrammed, hardware-initiated jump to a specific address. Occurs as a result of an error or certain other conditions.

tree-structured directories. A method for connecting directories such that each directory is listed in another directory except for the root directory, which is at the top of the tree.

truncate. To shorten a field or statement to a specified length.

TTY. Designates a terminal. On a system with more than one terminal, the TTY field of the process status displayed by the **ps** command indicates which terminal started the process.

typematic key. A key that repeats its function multiple times when held down.

typestyle. Characters of a given size, style and design.

Uid. See **user number**.

UNIX link. A mechanism that lets you use the **ln** command to assign more than one name to a file. Both the new name and the file being linked to must be in the same file system. A file is deleted when all the UNIX links (including the first link--the original name) have been removed. Synonym for **hard link**

update. An improvement for some part of the system.

user. The name associated with an account.

user account. See **account**.

user ID. See **user number**.

user list. A list, containing the user identification and access levels, of all operators who are allowed to use a specified file or library.

user name. A name that uniquely identifies a user to the system.

user number (Uid). A unique number identifying an operator to the system. This string of characters limits the functions and information the operator is allowed to use. The Uid can often be substituted in commands that take a user's name as an argument.

user profile. A file containing a description of user characteristics and defaults (for example, printer assignment, formats, group ID) to be conveyed to the system while the user is signed on.

user-replicated file system. A file system containing files and directories accessed only by specific users or for particular applications. These user files and directories are replicated on different sites in a cluster.

utility. A service; in programming, a program that performs a common service function.

V=F. Virtual=Fixed mode.

V=R. Virtual=Real mode.

V=V. Virtual=Virtual mode.

valid. (1) Allowed. (2) True, in conforming to an appropriate standard or authority.

value. (1) In Usability Services, information selected or typed into a pop-up. (2) A set of characters or a quantity associated with a parameter or name. (3) In programming, the contents of a storage location.

variable. A name used to represent a data item whose value can change while the program is running. Contrast with **constant**.

VCTC. Virtual Channel-to-Channel.

verify. To confirm the correctness of something.

version. Information in addition to an object's name that identifies different modification levels of the same logical object.

vi. A full screen editor.

virtual device. A device that appears to the user as a separate entity but is actually a shared portion of a real device. For example, several virtual terminals may exist simultaneously, but only one is active at any given time.

virtual machine (VM). A functional simulation of a computer and its related devices.

VM HPO. Virtual Machine High Performance Option.

VM HPO PMA. Virtual Machine High Performance Option Preferred Machine Assist.

VM/SP. Virtual Machine/System Product.

VM/XA. Virtual Machine/Extended Architecture.

VM/XA SP. Virtual Machine/Extended Architecture System Product.

virtual storage. Addressable space that appears to be real storage. From virtual storage, instructions and data are mapped into real storage locations.

virtual terminal. Any of several logical equivalents of a display station available at a single physical display station.

Volume ID (Vol ID). A series of characters recorded on the diskette used to identify the diskette to the user and to the system.

VTAM. Virtual Telecommunications Access Method.

wildcard. See **pattern-matching characters**.

word. A contiguous series of 32 bits (4 bytes) in storage, addressable as a unit. The address of the first byte of a word is evenly divisible by four.

work file. A file used for temporary storage of data being processed.

workstation. A device at which an individual may transmit information to, or receive information from, a computer for the purpose of performing a task, for example, a display station or printer. See **programmable work station** and **dependent work station**.

working directory. See **current directory**.

wrap around. Movement of the point of reference in a file from the end of one line to the beginning of the next, or from one end of a file to the other.

XA. Extended Architecture.

Special Characters

Numerics

A