digital

OpenVMS Software Overview

OpenVMS

Part Number: AA–PVXHA–TE

OpenVMS Software Overview

Order Number: AA–PVXHA–TE

May 1993

This manual summarizes the software capabilities of the OpenVMS operating system and describes computing environments in which the system operates.

Revision/Update Information:	This is a new manual
Software Version:	OpenVMS VAX Version 6.0

Digital Equipment Corporation
Maynard, Massachusetts

May 1993

ZK5974

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

Tables

Preface

The *OpenVMS Software Overview* presents an overall picture of OpenVMS software capabilities and computing environments. It describes how the OpenVMS operating system and related optional software function in diverse environments, including open environments in which OpenVMS operating systems are linked to other systems supplied by multiple vendors. The OpenVMS software features described in this overview indicate the variety of ways in which the system can be used.

This version of the manual summarizes the full capabilities of the OpenVMS VAX operating system and related software that run on the complete family of VAX processors. It also provides a brief introduction to the use of OpenVMS software on a series of new RISC processors based on the Alpha AXP architecture.

Intended Audience

This document is intended for all OpenVMS users—managers, analysts, developers, programmers, general users—and for anyone else who is interested in OpenVMS systems. Readers of this manual need not have any special knowledge of OpenVMS software products.

Document Structure

The *OpenVMS Software Overview* is divided into three parts, comprising seven chapters and one appendix. Each part of the manual presents one aspect of the overall picture of OpenVMS software capability.

Part I presents an introduction to the OpenVMS system, its software, and computing capabilities. It provides a brief summary of system capabilities, but does not include details of software design.

- Chapter 1 is a general introduction to OpenVMS systems, software, computing styles, and configurations, including multiplatform use. The chapter also discusses OpenVMS growth potential.
- Chapter 2 describes OpenVMS computing capabilities, including open, distributed, and high-integrity capabilities and manageability.

Part II provides a technical overview of the software available on an OpenVMS system that stands alone or is a member of a VAXcluster configuration.

- Chapter 3 describes the components in the base OpenVMS operating system.
- Chapter 4 discusses the OpenVMS software used for developing applications.
- Chapter 5 summarizes the user interfaces to the system.

Part III describes the use of OpenVMS systems in environments in which they are connected to systems supplied by Digital or by other vendors.

- Chapter 6 describes how OpenVMS systems fit in open, distributed computing environments that can include software from multiple vendors.
- Chapter 7 describes how OpenVMS systems support high-integrity, distributed, commercial-strength production system environments.

Appendix A lists the standards supported by OpenVMS software.

Associated Documents

This manual is an overview that can be read independently of other OpenVMS documentation. It refers to specific OpenVMS manuals related to certain topics. For a complete listing of OpenVMS manuals, see the *Overview of OpenVMS Documentation*, which lists all documentation for the OpenVMS VAX operating system and the OpenVMS AXP operating system.

The *OpenVMS Software Overview* discusses software capabilities of optional (layered) software products that run on the OpenVMS VAX operating system. For documentation that describes POSIX for OpenVMS, refer to the POSIX for OpenVMS documentation set. For information about Network Application Support (NAS) products, see the NAS documentation set. For information about specific PATHWORKS products, see the appropriate PATHWORKS documentation set. For information about other software that runs on the OpenVMS VAX operating system, refer to the appropriate product documentation.

Conventions

In this manual, every use of the term OpenVMS or VMS refers to the OpenVMS operating system.

The following conventions are also used in this manual:

italic text	Italic text emphasizes important information and indicates complete titles of manuals.
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, or the name of a file.
numbers	All numbers in text are assumed to be decimal, unless otherwise noted.
mouse	The term *mouse* refers to any pointing device, such as a mouse, a puck, or a stylus.

Part I
OpenVMS Computing

This part of the manual presents an overall picture of the OpenVMS system.

Chapter 1 introduces the software that runs on the system and the styles of computing that the system supports. It describes basic OpenVMS configurations, and the computing platforms and types of processors on which the OpenVMS system runs. The chapter also indicates the growth potential of OpenVMS systems.

Chapter 2 summarizes major kinds of OpenVMS computing capabilities:

- Open system capabilities
- Distributed system capabilities
- Production system capabilities
- System and network management capabilities

1 Introduction to OpenVMS Systems

The OpenVMS (Virtual Memory System) operating system is a highly flexible, general-purpose, multiuser system that supports the full range of computing environments. OpenVMS systems combine the high integrity and dependability of commercial-strength systems with the benefits of open, distributed systems.

OpenVMS operating systems can be integrated with systems from different vendors in open systems computing environments. Digital has "opened" the traditional VMS system to support software that conforms to international standards for an open environment. These industry-accepted open standards specify interfaces and services that permit applications and users to move between systems and allow applications on different systems to operate together.

The OpenVMS operating system configuration includes OpenVMS integrated software, software services and routines, applications, and networks. The system supports all styles of computing, from timesharing to real-time processing to transaction processing. OpenVMS systems configured with optional software support distributed computing capabilities and can function as servers in multivendor client/server configurations.

The OpenVMS operating system is designed to provide software compatibility across all processors on which it runs. The OpenVMS VAX system runs on the complete family of VAX (Virtual Address Extension) processors, encompassing a very broad range of computing power, from entry-level desktop workstations to midrange computers to high-performance systems. OpenVMS provides software compatibility across the whole family of VAX processor models. In addition, OpenVMS runs on a series of new RISC processors designed in compliance with the Alpha AXP architecture (as described in Section 1.4).

This chapter introduces the OpenVMS operating system, gives a general overview of the system software, and describes the various styles of computing that OpenVMS software supports. It summarizes the basic ways in which OpenVMS software can be configured and connected to other software, and discusses the hardware platforms and processors on which OpenVMS software runs. The chapter also describes the growth potential of OpenVMS systems.

1.1 What Is the OpenVMS Operating System?

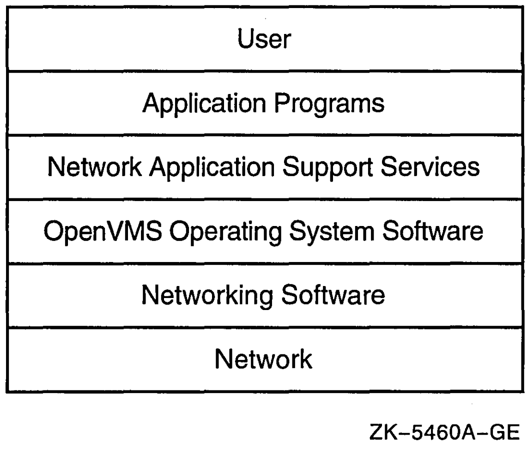
The OpenVMS operating system is a group of software programs (or images) that control computing operations. The base operating system is made up of core components and an array of services, routines, utilities, and related software. The OpenVMS operating system serves as the foundation from which all optional software products and applications operate. The services and utilities in the base OpenVMS operating system support functions such as system management, data management, and program development. Other integrated software that adds value to the system provides functions such as clustering and volume shadowing. (For a description of OpenVMS operating system software, see Chapter 3.)

Optional software products, including application programs developed by OpenVMS programmers and other programmers, run on the core operating system. The OpenVMS system supports a powerful, integrated development environment, with a wide selection of Digital software development tools and development tools supplied by other vendors. Application programs written in multiple languages supply computational, data processing, and transaction processing capabilities. Thousands of applications have been developed for OpenVMS systems by Digital and independent software vendors.

Networking software permits OpenVMS systems to communicate with other Digital systems and with systems supplied by other vendors in worldwide networks. Special-purpose networking software enables OpenVMS systems to function as powerful servers to personal computer clients in distributed client/server environments.

Optional Network Application Support (NAS) software that supports multivendor integration can be layered between the operating system and applications, as indicated in Figure 1–1. NAS run-time services provide standard application programming interfaces (APIs) to accomplish the interaction between applications and their users, data, systems, and other applications. NAS products, based on international standards, can also run on other standards-compliant systems. These products provide support for applications in distributed, multivendor environments. (For a discussion of how NAS software supplies open system capability, see Section 2.1.5.2.)

Figure 1–1 Software Running on an OpenVMS System


User
Application Programs
Network Application Support Services
OpenVMS Operating System Software
Networking Software
Network
ZK–5460A–GE

OpenVMS VAX software exhibits compatibility from version to version:

- User-mode programs and applications created under earlier versions of OpenVMS VAX run under subsequent versions with no change.
- Command procedures written under one version of OpenVMS continue to run under newer versions of the software.

OpenVMS software developed on VAX platforms can migrate easily to AXP platforms (see Section 1.4.1):

- Most OpenVMS VAX images run under translation on OpenVMS AXP systems.

- Most user-mode OpenVMS VAX sources can be recompiled, relinked, and run on an OpenVMS AXP system without modification. Only code that explicitly relies on the VAX architecture requires modification.

1.2 OpenVMS Styles of Computing

The OpenVMS operating system offers flexible, reliable support for all styles of computing. OpenVMS provides concurrent support for traditional modes of processing (batch, interactive, real-time) and multiple user interfaces (command line, windowing, and forms-based). In addition, OpenVMS can be easily tuned to support applications involving different computing styles: for example, timesharing, multiprocessing, and distributed client/server computing.

This section describes the varied styles of computing available to OpenVMS users. Chapter 2 summarizes computing capabilities supported by OpenVMS, including open, distributed, and commercially oriented capabilities. Chapter 6 and Chapter 7 describe the software that runs on OpenVMS systems in distributed multivendor environments.

1.2.1 Processing Modes and User Interfaces

In the traditional interactive mode of communication with the OpenVMS operating system, the user enters a command and the system executes it and responds. In the batch mode, commands to be executed by the operating system are placed in a file and submitted to the operating system for execution.

Real-time processes do not have to compete with interactive or batch jobs for scheduling priority within the OpenVMS operating system. A real-time process responds to events in related processes as they occur, rather than when the computer is ready to respond to them.

Standard user interfaces to the OpenVMS operating system include the character cell-interface and the windowing interface. Users with traditional character-cell terminals can interact with the command line interface to the system. Users with workstation terminals can access an optional graphical windowing interface, the DECwindows Motif interface. The user can employ point-and-click techniques to interact with menus and dialog boxes on multiple windows on the screen. DECwindows users can also invoke a window that emulates an OpenVMS terminal and can interact with the command line interface. Another possible user interface is a transaction interface, in which the user responds to a menu-driven forms interface customized for a particular application, such as an office application. Chapter 5 describes the various OpenVMS user interfaces.

1.2.2 Processing Styles

An OpenVMS system can be a single-user system or can provide timesharing functions for multiple users. In a timesharing configuration, many people can use the same computer at once. The computer interleaves time intervals for different user programs, performing a small portion of one user's program and then shifting to the next user. OpenVMS timesharing software is described in Section 3.1.1.

An OpenVMS system can be a uniprocessing or multiprocessing system. A uniprocessing system constitutes a single central processing unit (CPU) executing one copy of the OpenVMS operating system code and running one file system independent of other CPUs. In OpenVMS multiprocessing implementations, two or more processors are coupled together. Multiprocessing configurations include the following:

- The OpenVMS symmetric multiprocessing system (SMP) is a tightly coupled system in which all processors run the same copy of the operating system in memory. All processors can execute code from a single shared memory address space, dividing and sharing the workload. The processors are adjacent to each other, boot and shut down together, run a single file system, and appear as a single system from the system management and user points of view. OpenVMS SMP is described in Section 3.1.5.
- A VAXcluster system consists of a group of connected VAX CPUs. Each CPU can contain one or more processors. CPUs that are members of a VAXcluster system can share processing, mass storage, and other resources under a single management and security domain. Within this highly integrated environment, members retain their independence because they use local, memory-resident copies of the OpenVMS operating system. VAXcluster members can boot and shut down independently while benefiting from common resources. The VAXcluster is perceived by outside systems in the network as being a single system entity. VAXcluster configurations are described in Section 1.3.3 and VAXcluster software components in Section 3.4.1.
- A network is an example of a loosely coupled configuration. In a network, OpenVMS systems, VAXcluster systems, and other systems can be linked together, but the systems (nodes) can be separate, can be managed independently, and can run separate file systems. Networked systems are described in Section 1.3.4.

OpenVMS supports centralized and distributed processing capabilities across the full range of processors. Centralized processing involves having a single system control the use of computing resources and the execution of applications. OpenVMS provides the capability of controlling processors and their associated resources in a distributed manner as well as a centralized manner.

Distributed processing is the style of computing that permits portions of an application to be run on different systems, yet appear to the user as one integrated environment. The distributed application program can be divided into independent subsystems or modules that execute on different systems. The data and code for each module are separate, but the modules communicate with each other to share data or access to files or databases.

An example of one style of distributed computing is client/server computing: a client portion of an application on one system requests services from the server, which is usually on a different system.

For a description of OpenVMS distributed processing capabilities, see Section 2.2. Distributed computing environments are covered in Chapter 6.

1.3 Basic OpenVMS Configurations

OpenVMS systems are highly flexible and support a great variety of interconnections and configurations, including the standalone OpenVMS configuration, VAXcluster configurations, and network configurations involving OpenVMS systems. OpenVMS interconnections and configurations are described in the following sections.

1.3.1 Input/Output Connections

OpenVMS input-output (I/O) software supports connection to peripherals, communications devices, and networks. The I/O subsystem, which is part of the OpenVMS operating system kernel, processes I/O requests to external devices (as described in Section 3.1.1).

OpenVMS I/O provides access to storage buses and interconnects and system-dependent open buses (open buses are those for which any vendor can obtain specifications and build adapters). The following kinds of equipment can be attached to buses and interconnects supported by OpenVMS: conventional disk and tape devices, compact disc devices, other terminal and workstation connectors, and communications adapters.

OpenVMS I/O provides for connection to other systems, devices, and local area networks (LANs) by means of LAN adapters. The I/O subsystem also supports connection to protocols that permit communication with other systems over DECnet and TCP/IP networks (as described in Section 1.3.4).

Table 1–1 lists examples of some of the buses, interconnect devices, and LAN adapters supported by OpenVMS I/O.

Table 1–1 Examples of Buses, Interconnects, and LAN Adapters Supported by OpenVMS Systems

Device	Description
Storage Buses and Interconnects	
Computer interconnect (CI) bus	Very high-speed, dual-path bus that joins VAX computers and intelligent I/O subsystems.
Digital Storage System Interconnect (DSSI) bus	High-speed storage bus that permits multiple computers to communicate directly with storage devices.
Small Computer System Interconnect (SCSI)	Interconnect that permits devices complying with the ANSI SCSI standard to be connected to small VAX computers.
Open Buses and Interconnects	
TURBOchannel I/O interconnect	Open bus for desktop applications, used on desktop VAXstation systems; provides the high performance required to support advanced computing systems involving complicated I/O options (for graphics, imaging, high-speed networking and data collection, and large database storage). TURBOchannel is a Digital open standard; the specifications are available to other vendor designers.
VMEbus	High-performance open I/O bus used on certain mid-range processors that supports device drivers written by other vendors and can act as a system interconnect from multiprocessing systems.
OpenVMS SCSI implementation	Open interconnect that supports SCSI-compliant other-vendor devices and user-written device drivers.

(continued on next page)

Table 1–1 (Cont.) Examples of Buses, Interconnects, and LAN Adapters Supported by OpenVMS Systems

Device	Description
	LAN Adapters
Ethernet (IEEE 802.3) adapter	Communications device that provides a reliable, low-cost, coaxial cable connection.
Fiber Distributed Data Interface (FDDI) adapter	Communications device that provides a connection to a high-performance fiber optic interconnect.
Token ring (IEEE 802.5) LAN adapter	Communications device that provides a connection to a baseband token ring LAN (used in PATHWORKS configurations).

For communication with personal computers and Macintosh systems in a PATHWORKS environment (as described in Section 2.2.2.2), OpenVMS servers can connect to Ethernet or token ring LANs. For communication with devices such as terminals, modems, and printers offered by terminal servers on a LAN, OpenVMS systems support connection to the local area transport (LAT). OpenVMS also supports connection to the local area disk (LAD) protocol for access to compact disc media (CD–ROMs) that reside on Digital InfoServer systems. In addition, the I/O software permits direct and remote connection of monitors and terminals and windowing workstations.

1.3.2 Standalone System Configurations

A standalone OpenVMS system operates independently, not linked to other systems. The basic hardware configuration comprises the following:

- The CPU: The hardware that handles all computing operations, calculating and routing input and output and executing programs. A standalone system can be a uniprocessor system supporting one processor or a multiprocessor system supporting two or more processors.
- The memory modules: A series of physical locations that store data and instructions in binary form.
- I/O: Device drivers (a combination of hardware and software), peripheral devices, and buses and interconnects.

The OpenVMS software running on the standalone hardware platform can be capable of handling timesharing activities for multiple users as well as executing multiple individual programs for single users. OpenVMS standalone capabilities are described in Part II of this manual. Section 6.1 compares capabilities supported by OpenVMS in standalone and distributed environments.

1.3.3 VAXcluster Configurations

A VAXcluster system is a highly integrated OpenVMS computing environment: a connected group of VAX systems that share resources under a single OpenVMS management domain. Shared resources include image and data files, batch queues, print queues, disk and tape storage, and system management.

The ability to survive failure of any single component makes the VAXcluster system suitable for developing high-availability applications such as transaction processing applications. VAXclusters also make effective servers for clients on personal computers and other desktop systems (see Section 2.2.2.2).

To users and applications, the VAXcluster system functions as a single system, except that applications and data continue to be available to users on remaining members (nodes) of the VAXcluster system even if a member shuts down. Within the VAXcluster configuration, each member can boot or shut down independently. If one member shuts down, the user can log in to another member of the cluster and continue working. Because disk and tape storage is shared across the VAXcluster, the user can continue to access the original data and applications. Individual members of the VAXcluster system can be serviced without interruption to applications running on the VAXcluster.

VAXcluster systems can make disk and tape resources available to cluster nodes. Through disk and tape server software on the VAXcluster, cluster-accessible disks and tapes can be accessed simultaneously by multiple active nodes in the VAXcluster. Cluster-accessible disks permit high-performance, clusterwide read/write file sharing. Because computers can share a single version of a file, updates to a file need be made only once. Clusterwide queues permit sharing of batch processing and printer resources. Batch and print queues can be processed on any node that has access to the disk.

The VAXcluster system is a combination of hardware and software. VAXcluster hardware includes a variety of buses and other connection devices as well as controllers for resource sharing. Hardware resources can be expanded incrementally or gradually, as necessary. VAXcluster software includes drivers, controllers, and traffic managers (as described in Section 3.4.1). VAXcluster processors, controllers, disks, and tapes can be added or removed without requiring that the VAXcluster be shut down.

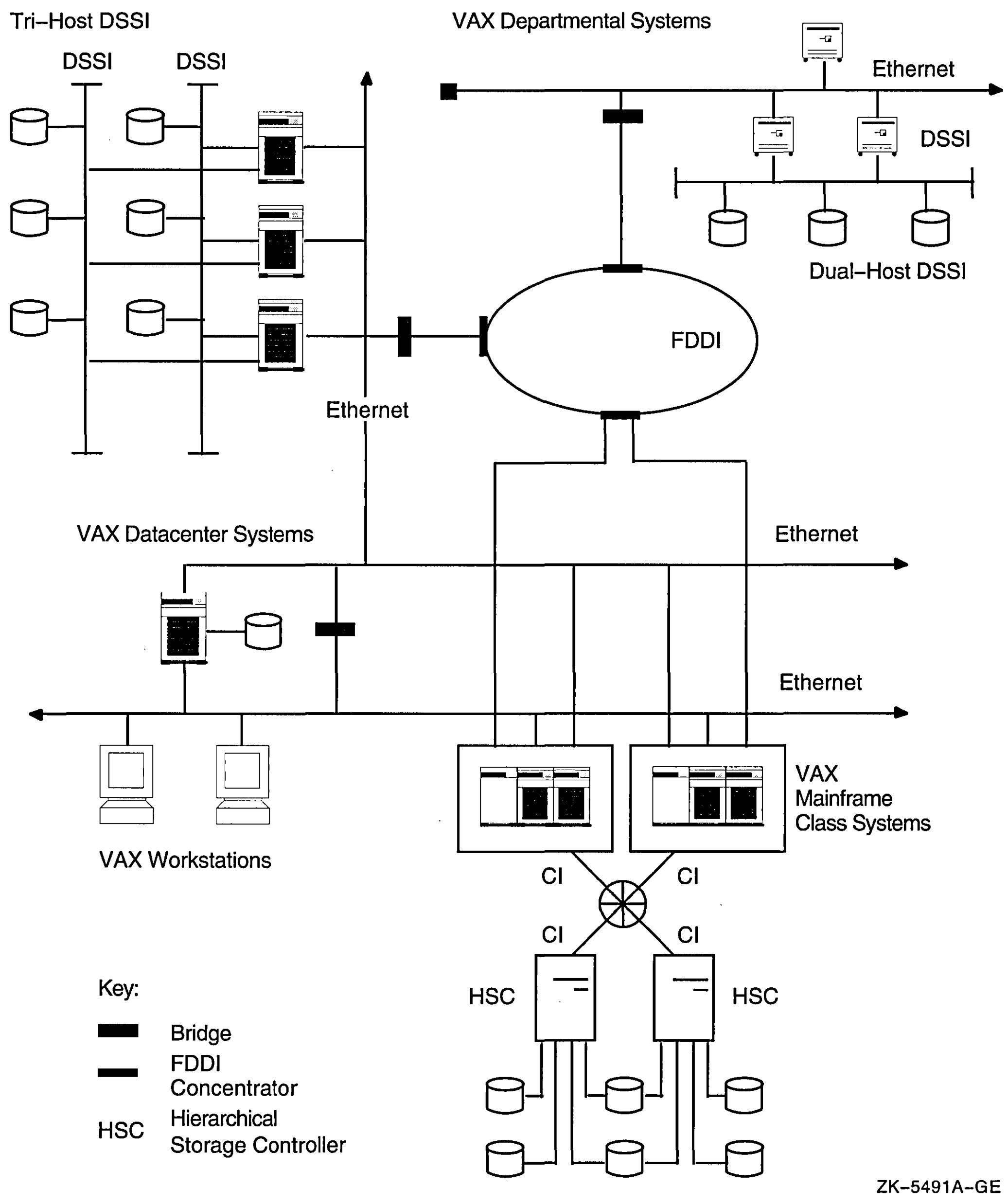
VAXcluster configurations are highly flexible and support the full range of VAX processors with the OpenVMS operating system. In a VAXcluster system, two or more CPUs are connected by means of communications media, called interconnects, which the VAXcluster software uses for Systems Communications Architecture (SCA) I/O traffic. The VAXcluster members use interconnects to communicate with each other. VAXcluster software supports any combination of the CI, DSSI, Ethernet, and FDDI interconnects (described in Table 1–1).

A VAXcluster configuration that includes a CI can optionally be configured with hierarchical storage controller (HSC) units that let nodes connected to the CI share disks. Each member of a multihost DSSI VAXcluster can share all disks attached to the DSSI bus.

The Ethernet and FDDI are industry-standard general-purpose communication interconnects that may be used to implement a LAN. A VAXcluster system that uses a LAN interconnect is called a local area VAXcluster. The FDDI can also be used in a disaster-tolerant VAXcluster configuration.

A VAXcluster system that uses more than one type of interconnect for VAXcluster communication is referred to as a mixed-interconnect VAXcluster system. An example of a mixed-interconnect VAXcluster configuration is given in Figure 1–2.

Figure 1–2 Mixed-Interconnect VAXcluster Configuration



1.3.4 Networked Systems

Networking software and hardware products permit OpenVMS operating systems to communicate with other systems. The network consists of computer systems connected by physical means, such as cables or microwave links. Optional networking software running on the OpenVMS system supports networking protocols (sets of rules and operating procedures) that permit communication between Digital and other vendor systems in worldwide networks.

DECnet and TCP/IP networking products that run on OpenVMS systems are listed in Table 1–2. In addition, DECnet, TCP/IP, and AppleTalk network transports permit OpenVMS servers to communicate with personal computer and Macintosh clients in PATHWORKS environments. Use of networking products in multivendor distributed computing environments is summarized in Chapter 6.

Table 1–2 Networking Software Products That Run on OpenVMS Systems

Networking Product	Protocols Supported
	DECnet Software
DECnet/OSI for OpenVMS VAX	OSI protocols, permitting communication with any vendor's system that supports standards for Open Systems Interconnection (OSI)
	Digital Network Architecture (DNA) protocols, permitting communication with any system running DECnet/OSI or DECnet Phase IV software on the DECnet network
DECnet for OpenVMS	DECnet Phase IV DNA protocols, permitting communication with any system running DECnet Phase IV software on the DECnet network
	TCP/IP Software
DEC TCP/IP Services for OpenVMS	The TCP/IP protocol, permitting communication with systems based on UNIX and other systems on the Internet
DEC NFS for OpenVMS	The Network File System (NFS) protocol, permitting UNIX clients to access OpenVMS files and files compatible with UNIX on OpenVMS systems

DECnet is the traditional Digital networking product, supported on most Digital systems, including the OpenVMS operating system and operating systems based on UNIX (ULTRIX and DEC OSF/1). DECnet software provides peer-to-peer networking. DECnet software on the OpenVMS system has proven its reliability over the years in extensive networking configurations, including the very large Digital network.

DECnet/OSI (the latest phase of DECnet) conforms to international standards for the Open Systems Interconnection (OSI) model (as described in Section 2.1.5.1) and supplies open, multivendor networking capabilities. DECnet/OSI is compatible with the previous phase of DECnet, Phase IV. This compatibility between phases preserves the investment in networking capabilities made by DECnet users on OpenVMS systems.

DECnet/OSI for OpenVMS VAX is the optional networking product that integrates OSI and DECnet Phase IV capabilities and includes additional capability. The DECnet for OpenVMS product (formerly called DECnet–VAX Phase IV) continues to be supported for OpenVMS VAX Version 6.0.

DECnet systems can be connected to either local area networks (LANs) or wide area networks (WANs). A LAN is a network of systems in a specific geographical area, while a WAN permits long-distance communications. The two kinds of networks can be integrated into a single network.

High-speed LAN communications occur over Ethernet/IEEE 802.3 or FDDI interconnections. Ethernet networks are flexible, reliable, and low cost. The FDDI fiber optic interconnect provides 10 times the transmission rate of the Ethernet, and can seamlessly interoperate with an Ethernet on a LAN extended by means of a LAN bridge. A third LAN technology, token ring, is supported in PATHWORKS configurations. The token ring LAN is a baseband ring network specified by IEEE 802.5 and is compatible with IBM 802.5/token ring networks.

DECnet/OSI products available on OpenVMS systems include WAN software for connection to X.25 packet-switching data networks (for example, TYMNET) that conform to international recommendations and gateway software for connection to other networking products (for example, IBM SNA interconnect products). DECnet/OSI also supplies distributed name service and time service software that complies with open standards for distributed computing environments (see Section 2.2.5).

OpenVMS users can connect to Internet networks using the DEC TCP/IP Services for OpenVMS product (which is the latest version of the VMS/ULTRIX Connection product). TCP/IP is the de facto standard for networking UNIX systems (see Section 6.2.2). The Internet is an openly accessible worldwide research and commercial network.

OpenVMS systems can also act as servers to personal computer clients (such as DOS, OS/2, and Macintosh clients) over network connections, using appropriate PATHWORKS software on each client and server. OpenVMS servers support Ethernet or token ring LAN connections, TCP/IP connections, and AppleTalk connections. The PATHWORKS environment is introduced in Section 2.2.2.2 and described in detail in Section 6.4.2.

Figure 1–3 illustrates a multivendor open network that includes DECnet/OSI, DECnet Phase IV, and TCP/IP systems as well as other vendor systems. Routing is provided by multiprotocol wide area networking routers, including the DEC WANrouter and the DECnet Network Integration Service (DECNIS) router.

1.4 OpenVMS Systems on Multiple Platforms

The OpenVMS operating system is available on two hardware platforms:

- A complex instruction set computer (CISC) architecture based on the VAX architecture
- A reduced instruction set computer (RISC) architecture based on the Alpha AXP architecture

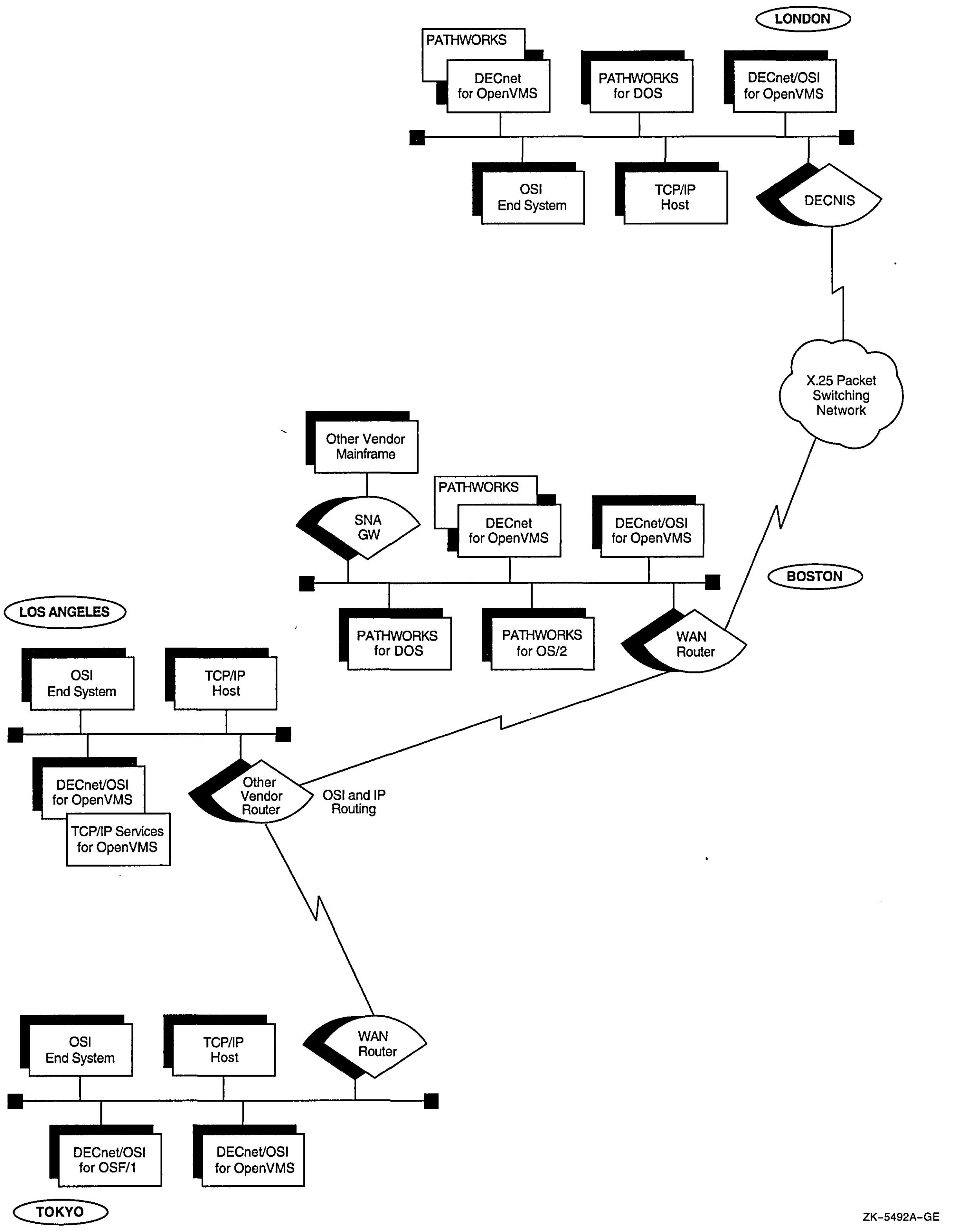
The OpenVMS VAX operating system runs on the CISC platform, and the OpenVMS AXP operating system runs on the RISC platform. From the user's point of view, the OpenVMS AXP operating system is a port of the OpenVMS VAX operating system, not a redesign of the operating system.

A computing architecture provides a set of rules that determine how the computer actually works; for example, the architecture defines how the data is addressed and the instructions are handled.

The VAX architecture is a flexible CISC architecture that provides for 32-bit processing. The instruction set consists of a large number of instructions of variable length, some of which are complex. The 32-bit VAX architecture provides sufficient address space for current and future applications. The capabilities of OpenVMS on VAX processors will continue to be expanded.

The VAX architecture is used across the entire VAX family of systems. The use of a single, unifying, integrated VAX architecture with the OpenVMS software permits an application to be developed on a VAX workstation, prototyped on a small VAX system, and put into production on a large VAX processor or run on a fault-tolerant VAXft processor. The advantage of the VAX system approach is

Figure 1–3 Multivendor Network Topology



that it enables individual solutions to be tailored and fitted easily into a larger, enterprisewide solution. The hardware design of VAX processors is inherently reliable and is particularly suitable for high-availability applications (such as dependable applications for mission-critical business operations and server applications for a wide variety of distributed client/server environments).

The Alpha AXP architecture is a high-performance RISC architecture that can provide 64-bit processing on a single chip. The instruction set consists of uniformly sized simple instructions that can be executed rapidly. The 64-bit capability is especially useful for applications that require high-performance and very large addressing capacity. A move to the AXP design from a VAX processor can take place whenever applications require the performance or capacity. (For example, AXP processors are particularly appropriate for graphics- or numeric-intensive software applications that involve imaging, multimedia, visualization, simulation, and modeling.)

The use of the RISC architecture in AXP processors offers increased CPU performance. The Alpha AXP architecture will run OpenVMS on a full range of single processors, multiprocessors, and parallel processors (see Section 1.4.2).

1.4.1 System Compatibility and Program Portability Across Platforms

The OpenVMS AXP operating system is compatible with OpenVMS VAX Version 5.4–2 systems in terms of user and programming environments. For general users and system managers, OpenVMS AXP has the same interfaces as OpenVMS VAX. For programmers, the software development environment involves the same programming interfaces and development tools as with OpenVMS VAX. Most OpenVMS VAX application programs can be recompiled and relinked and then run on OpenVMS AXP. OpenVMS VAX and OpenVMS AXP systems can coexist in the same networking environment.

Programs that run on OpenVMS VAX systems can be converted to run on OpenVMS AXP systems by recompiling and relinking or by translating. A single application can include both native images (those that were recompiled and relinked) and translated images.

The most effective way to convert a program from OpenVMS VAX to OpenVMS AXP is to recompile the source code using a native OpenVMS AXP compiler and then to relink the object files and shareable images using the OpenVMS Linker. This method produces a native OpenVMS AXP image that can take full advantage of the speed of the AXP system.

The alternative method, translation, involves using the VAX Environment Software Translator (VEST) to translate user-mode OpenVMS VAX images to run on OpenVMS AXP systems. The translated image provides a higher degree of OpenVMS VAX compatibility, but does not provide the same high performance as a recompiled image. Translation is used primarily when recompilation is not practical or possible.

For current information about available OpenVMS AXP capabilities, refer to the *OpenVMS AXP Version 1.5 Upgrade and Installation Manual* and the *OpenVMS AXP Version 1.5 Release Notes*.

1.4.2 Processors on Which OpenVMS Systems Run

VAX processors and AXP processors can be used to run OpenVMS software:

- The OpenVMS VAX operating system runs on the full family of compatible VAX processors: desktop, deskside, departmental, datacenter, and mainframe-class computers, as well as fault-tolerant and real-time computers.
- The OpenVMS AXP operating system runs on AXP processors that also range from low- to high-end computers, including desktop and deskside workstations and servers, departmental systems, and datacenter and mainframe systems.

In the categories of desktop and deskside workstations and servers, VAX computers offer increasing levels of performance. Low-end VAX desktop systems have graphics capability and can handle office applications. Other VAX desktop computers offer more resources and graphics capabilities and use open buses. The highest performance VAX departmental systems operate in an open office environment and can handle virtually any distributed computing requirements.

Low-end AXP desktop processors feature accelerated graphics and use an open interconnect. AXP deskside workstations and servers provide very high performance capabilities. AXP departmental processors are suitable for demanding office applications, offering many options for commercial workloads and batch operations.

In the midrange category, VAX computers suited for critical datacenter use provide very high performance capabilities in transaction processing. Some uniprocessors can be expanded by the addition of up to five processors to achieve six-way symmetric multiprocessing (SMP). Datacenters can group midrange computers together in VAXclusters to supply higher performance for demanding applications.

The VAX 7000 midrange computer can optionally be converted to an AXP machine. AXP processors used in datacenters are versatile machines that can provide extremely high performance in support of a wide range of general-purpose applications, capable of accepting requests for computing services from many clients at the same time and handling mixed workloads.

VAX computers that meet high-end computing needs include mainframe systems suitable for commercial applications and supercomputer models suitable for scientific and technical computing. Both supercomputers and mainframes also have server models that feature a shareable database and connectivity to workstations and personal computers. VAX supercomputing servers are available with up to four symmetric multiprocessing units, each with a vector processor to meet demanding loads for scientific and engineering networks. All models can be integrated with and are compatible with the entire VAX system family.

One high-end model, the VAX 10000, can optionally be converted to an AXP machine. The mainframe-class of AXP processors are very high performance systems, capable of solving the most critical computing needs of worldwide industries and businesses. The flexible, modular design permits upgrading of memory, I/O, and system interconnects.

The VAXft computer is a fault-tolerant processor designed for critical applications that cannot tolerate down time. The VAXft computer guarantees continuous processing with the highest levels of data and computational integrity. The VAXft computer maintains continuous computing operations because it is designed to be completely redundant; the hardware is separated into two zones, each containing duplicate modules and cross-checking capabilities. The VAXft design ensures that there is no single point of hardware failure.

The rtVAX systems, which support the VAXELN real-time operating environment, are specially optimized for real-time applications built on VAX processors running OpenVMS. VAXELN software runs on rtVAX systems, including systems that support high-performance real-time use.

1.5 OpenVMS Growth Potential

OpenVMS managers and users can expand and alter their computing capabilities without risk of obsolescence or system-related constraints. In addition to being scalable for the full range of processors from desktops to mainframes, the OpenVMS operating system offers the flexibility to move the software from VAX platforms to AXP platforms as processing requirements grow or change. And OpenVMS distributed multivendor system capabilities give managers the opportunity to integrate systems and devices from multiple vendors into the OpenVMS computing environment and to integrate OpenVMS software into other vendor's environments.

OpenVMS provides protection of computing resources against obsolescence and incompatibility and ensures investments in applications and data across systems. The system also supports new technologies and standards as they are developed, without sacrificing any existing capabilities. Over the years, new capabilities such as VAXcluster systems, multiprocessing, fault tolerance, and disaster tolerance have been integrated in the OpenVMS environment. All the while, programs compiled and linked on the earliest VAX processors run, without change, on the newest VAX processor, using the latest version of the OpenVMS operating system; these same programs can also migrate easily to run on AXP processors. In addition, the compatibility of the OpenVMS operating system on all processors protects the investment in application development and training. It is possible to add capacity without having to revise applications or retrain people.

OpenVMS offers multiple growth options. OpenVMS customers can upgrade or replace existing capabilities, group systems into VAXclusters or add to existing VAXcluster configurations, or link systems through networking connections.

Upgrades of processor hardware do not involve a long, complicated system generation process. The OpenVMS operating system boots on the processor or workstation, automatically configuring itself to the CPU, memory, and devices present. Installation of the OpenVMS software is simplified through the use of the OpenVMS installation procedure. The user can upgrade to new software releases or add layered software with minimal disruption.

OpenVMS is highly flexible and can accommodate to change readily. As the configuration of the OpenVMS system is modified through the addition or subtraction of systems or changes in vendors' devices, the OpenVMS system will continue to work in a reliable, predictable manner. VAXcluster design permits the addition of multiple processors, storage controllers, disks, and tapes without requiring that the VAXcluster system be shut down. OpenVMS distributed computing capabilities permit systems and resources to be added without disruption of the network. In addition, with OpenVMS support for open interfaces, managers can determine what open systems and other capabilities are required to meet their needs. OpenVMS standards-based open networking capabilities permit free interconnection of systems from different vendors. Using NAS software, applications can move easily across different systems in a network and have the systems all work together as if they came from one vendor.

2 OpenVMS Computing Capabilities

The OpenVMS operating system provides an array of capabilities that support the full range of computing environments. A computing environment is made up of resources that are compatible with each other and all work together toward a common goal. In general, OpenVMS environments can supply the following kinds of capabilities (which can exist in any combination):

- Open system capabilities
- Distributed processing capabilities
- Production system capabilities
- System and network management capabilities

OpenVMS software capabilities include both the standardized features of open systems computing and the commercial-strength functionality of traditional OpenVMS systems. System and network management software provides for control of heterogeneous, integrated environments.

This chapter describes the capabilities supported in OpenVMS computing environments and summarizes the software resources available in each environment.

2.1 Open Systems Capabilities

OpenVMS offers the benefits of an open system environment, which permits both applications and users to move between systems. In addition, applications on different open systems can operate together.

The OpenVMS operating system makes available a set of services in an open domain, while still offering its traditional high-integrity computing services. Incorporation of open computing capabilities enhances the traditional feature-rich OpenVMS environment.

2.1.1 What Is an Open System?

The Institute of Electrical and Electronic Engineers (IEEE) Computer Society has developed a definition of an open system that Digital has adopted. This definition is based on standard interfaces rather than on standardization of products. The IEEE definition of an "open" system is as follows:

> "A system that implements sufficient open specifications for interfaces, services, and supporting formats to enable properly engineered applications software:
>
> - To be *ported* across a wide range of systems with minimal changes
> - To *interoperate* with other applications on local and remote systems, and
> - To *interact* with users in a style which facilitates user portability."

The IEEE definition uses the term “open specification”: “A public specification that is maintained by an open, public consensus process to accommodate new technology over time and that is consistent with standards.”

Open specifications do not rely on any particular technology or product; they allow users to determine what open systems and other capabilities are required to meet their needs.

Software in the OpenVMS open systems environment enables the development and use of portable applications and consistent user interfaces and also permits systems to operate together. The keys to openness of OpenVMS systems are standard programming interfaces, standardized user interfaces, and standard protocols.

2.1.2 Support of Open Standards and Specifications on OpenVMS Systems

Digital is active in standards bodies that specify or adopt critical standards for an open environment. As part of this effort, Digital has “opened” VMS to conform to the majority of open, international standards for portability, interoperability, and a consistent user interface. Some of the open standards and specifications supported by OpenVMS are:

- IEEE standards and draft standards for POSIX, the “Portable Operating System Interface”
- X Open standards according to the X/Open Portability Guide Issue 3 (XPG3) BASE specification
- Standards developed by the Open Software Foundation (OSF) for:
  - Application Environment Specification (AES)
  - Distributed Computing Environment (DCE)
  - Distributed Management Environment (DME)
- American National Standards Institute (ANSI) accredited standards for languages
- The International Organization for Standardization (ISO) standards for the Open Systems Interconnection (OSI) model

Additionally, Digital participates with the National Institute of Standards and Technology (NIST, formerly NBS) of the U.S. Government in its workshops and supports NIST’s work in defining an environment, based on formal standards, in which to develop and support portable applications. Digital is also certified by NIST as an accredited U.S. Government OSI Profile (GOSIP) Test Facility.

POSIX for OpenVMS VAX has passed the NIST-developed POSIX Conformance Test Suite (PCTS) and has been granted a certificate of validation for conformance to Federal Information Processing Standards Publication 151-1 (FIPS 151-1) by NIST.

The OpenVMS operating system has received X/Open BASE-level branding for the X/Open Portability Guide Issue 3 (XPG3) environment, as specified by the X/Open consortium. The X/Open consortium of major vendors is establishing a Common Applications Environment based on formal standards and other widely accepted specifications.

The XPG3 BASE brand confirms that the OpenVMS system provides the open systems functionality specified by XPG3. BASE-level branding applies to the operating system and covers:

- System interfaces (includes POSIX 1003.1 plus extensions)
- Commands and utilities: A shell environment (includes POSIX P1003.2 with extensions)
- Internationalization features

For a summary of standards and specifications that OpenVMS supports, refer to Appendix A.

2.1.3 Application Portability

Application portability is the capability to move an application from one system to another easily. Standard programming interfaces permit application and data portability. Portable applications written strictly to a suite of open specifications provide the following benefits:

- Applications can be written once and run on other open platforms that support the standards used in the applications.
- Users can access the wide range of applications available on open platforms.
- Applications are vendor independent.

Applications that are developed on the OpenVMS system and conform to open standards can be moved to other systems that conform to the same standard interfaces. OpenVMS POSIX applications can be ported to other platforms that support the same POSIX and X/Open standards and draft standards, including many UNIX platforms. Applications written in ANSI and ISO languages are portable to other systems. The Structured Query Language (SQL) supports data portability between the OpenVMS environment and other environments. In addition, the OSF/Motif graphical user interface supports application portability.

The following sections describe each of these capabilities.

2.1.3.1 OpenVMS POSIX Application Portability

POSIX standards enable users and conforming applications to move between systems supporting the POSIX standards. On the OpenVMS system, POSIX and X/Open standards and draft standards are available together with the OpenVMS environment, thus allowing portable applications to benefit from the proven features of OpenVMS.

IEEE POSIX standards are a set of implementation-independent system interface standards. IEEE POSIX standards have the "look and feel" of UNIX interfaces. To date, the following standards and draft standards are supported by POSIX for OpenVMS VAX:

- 1003.1 (standard): Specifies basic system services of an operating system, allowing development and operation of portable applications. (This standard covers required data structures, system primitives, and library interfaces).
- P1003.2 (draft): Specifies shells and utilities, providing additional services for installation, execution, and user portability (for example, command syntax and command parameters).
- P1003.4 (draft): Specifies extensions for developing real-time applications and extends the services provided in 1003.1.

The availability of POSIX P1003.2 and P1003.4 distinguishes OpenVMS POSIX from many other proprietary operating systems, which often limit their standards offering to 1003.1.

POSIX for OpenVMS also supports X/Open standards specified in the XPG3 BASE specification. (OpenVMS POSIX XPG3 features are described in Section 4.4.2.)

An application that strictly conforms to POSIX and X/Open standards and draft standards can be developed on an OpenVMS system with OpenVMS POSIX and then ported without modification to any other platform that supports the same POSIX and X/Open standards or draft standards. Similarly, an application that strictly conforms to POSIX standards that is developed on a platform other than OpenVMS can be ported to and run on an OpenVMS system on which POSIX for OpenVMS VAX is installed. (OpenVMS POSIX programming is described in Section 4.4 and OpenVMS POSIX user interfaces in Section 5.3.)

POSIX applications and users on OpenVMS can access and benefit transparently from the feature-rich OpenVMS environment in addition to the standard interfaces. Some of the key benefits directly available for POSIX users on OpenVMS are OpenVMS security features, volume shadowing, clustering, and many features of the software development environment. The availability and use of POSIX for OpenVMS does not affect other regular OpenVMS users.

2.1.3.2 Other Application Portability Features

Applications need a standard way to interface with the variety of databases available in open environments. The data language of choice for applications is SQL, the ISO and ANSI language standard that OpenVMS supports through the DEC Rdb database product. Applications based on relational database management systems can use SQL as the language for defining and updating and querying in the database.

Applications written in ANSI/ISO languages are easily portable to other platforms that support them. OpenVMS provides support for such languages as Ada, BASIC, C, COBOL, FORTRAN, LISP, and Pascal.

2.1.3.3 OSF/Motif Applications

OSF/Motif, a component of the OSF Application Environment Specification (AES), is an industry standard for an open graphical user interface and its associated application programming interface (API). The OpenVMS graphical user interface is DECwindows Motif. Any Motif application can run on DECwindows Motif.

2.1.4 User Portability

User portability relates to the ease with which users can move between applications, experiencing a consistent user interface on different systems. Standard services and interfaces incorporated in open applications enhance user portability. In addition, user portability tools that are based on standards permit development of consistent user interfaces. Standards-based tools allow end users, developers, and managers to move easily from tool to tool, eliminating the need for retraining.

Standards-based user interface software available in the OpenVMS open systems environment includes OSF/Motif and the OpenVMS POSIX shell and utilities.

2.1.4.1 OSF/Motif User Interface

The OSF/Motif interface implemented by DECwindows Motif provides users with a consistent screen appearance and behavior for applications that conform to the X Window System. (For a description of the DECwindows Motif graphical user interface, see Section 5.2.2.)

2.1.4.2 POSIX Shell and Utilities

The POSIX interface in the OpenVMS operating system allows users to develop and run applications that can also be run on any other platform that supports POSIX and X/Open standards and draft standards. POSIX P1003.2 defines the shell and utilities services that comprise a user interface environment similar to the Korn shell in the UNIX environment. Included in the interactive shell are commands similar to UNIX commands. OpenVMS users running OpenVMS POSIX can experience the look and feel of UNIX. (For a description of the OpenVMS POSIX user interface, see Section 5.3.)

2.1.5 Multivendor Interoperability

Interoperability is the ease with which systems from multiple vendors can work together, sharing data and integrating applications. Interoperability and integration of systems from different vendors involves connecting the systems. All systems and their resources are widely available in an open environment. In addition, an open system interoperates with any installed computing environment.

Multivendor interoperability provides the following benefits:

- Permits easy interconnection of different multivendor systems throughout the enterprise to communicate and share data and applications
- Provides the flexibility to change and expand the computing environment to meet changing needs
- Allows unimpeded access to systems and use of data

In an OpenVMS environment, applications can interoperate with different applications running on other systems that support the same standards. OpenVMS supports the capabilities supplied by networking software and protocols, NAS software for integrating applications on different vendor systems, and DCE services for a distributed computing environment. The following sections describe these capabilities.

2.1.5.1 Open Networking Capability

OpenVMS networking software that provides open networking capability includes the DECnet/OSI for OpenVMS VAX product, based on ISO standards for the Open System Interconnection (OSI) model. The OSI model allows free interconnection of systems from different vendors in an open systems network. Participation in the network is open to any vendors that conform to the appropriate ISO standards. DECnet/OSI for OpenVMS VAX supports OSI protocols and DNA protocols.

In addition, DEC TCP/IP Services for OpenVMS, an optional software product, supports connection to UNIX systems on the Internet. The Network File System (NFS) server software on OpenVMS allows UNIX clients to access both the OpenVMS file system and files compatible with UNIX on the OpenVMS system.

These networking products are summarized in Section 1.3.4. Distributed networking capabilities are discussed in Section 2.2.5, and the distributed heterogeneous networking environment is covered in Section 6.2.

2.1.5.2 Network Application Support (NAS) for Open Systems

Network Application Support (NAS) products are based on open international standards and specifications, including the AES and DCE standards developed by OSF. NAS software (called "middleware" because it is layered between the operating system and applications) provides application programming interfaces (APIs) to accomplish the dialog between applications and their users, data, systems, and other applications.

NAS is the Digital implementation of an open systems profile that provides presentation, communication, control, information, computation, and management services. NAS software provides for interoperability between applications in an open system environment (see Section 2.2.3). NAS products can be used to integrate applications across a distributed, multivendor environment (see Section 6.3).

Other interoperability capabilities are provided by OSF DCE services. The OSF DCE consists of a set of basic mechanisms that allow application developers to design interoperable applications. (DCE technologies are covered in Section 2.2.4 and the DCE software environment is described in Section 6.3.2.)

2.2 Distributed Computing Capabilities

In a distributed computing environment, an application is distributed over two or more systems or processors, each of which has its own autonomous operating environment. A distributed application is composed of separate modules, running on different systems, that communicate with each other by passing data between modules or by sharing access to files or databases. A distributed application must be able to coordinate its activities over a dispersed operating environment. (Distributed and centralized styles of computing are contrasted in Section 1.2.2.)

The distributed computing environment can consist of software located in a single box or single room or can comprise a worldwide network of computers. The systems in the distributed configuration can be uniprocessor, multiprocessor, or VAXcluster systems; systems from different vendors can be included in the same configuration.

2.2.1 Client/Server Style of Computing

One style of distributed computing that permits resource sharing between different systems is client/server computing. In the client/server environment, portions of an application are distributed across the network between servers and clients:

- A server is any system that provides a service or resource to other systems.
- The client is the system requesting the service.

This style of computing allows each portion of a distributed application to run in its own optimal environment. The whole application does not have to run on one centralized system (such as a mainframe system), but enterprisewide cohesiveness can still be maintained. For example, individuals or local offices, using their own computers and running software appropriate to their needs, can be linked to large computers or VAXcluster systems in a network. A distributed computing system can function as though it were a single system that connects all parts of an enterprise. The client can have transparent access to the integrated resources of the enterprise.

Any system can be a client or a server, and some systems may include both client software for certain applications and server software for other applications. Servers can be connected to many clients, and a client can be connected to more than one server at a time. (Client and server relationships may change frequently: at times it may not be possible to tell which is the client and which is the server.) In some cases, the application is stored on the server and run on the client, using the resources of the client. The user, who does not need to know what system is serving the application, can function in a familiar, local environment.

2.2.2 OpenVMS Client/Server Capabilities

OpenVMS systems support a wide variety of client/server configurations. Clients requiring resources can be personal computers, workstations, point-of-sale devices, OpenVMS systems, or other vendor systems that are running the appropriate client software. User interfaces on client systems can be character-cell terminals or windowing desktops.

Servers fulfilling clients' requests can be located on OpenVMS systems or other operating systems running appropriate server software. OpenVMS servers, for example, can provide file access, printing, application services, communication services, and computing power as application engines to clients on desktop devices or in laboratories or factories. Client/server configurations permit the commercial-strength capabilities of OpenVMS host systems to be integrated with the personal-computing capabilities of desktop systems.

NAS software, which runs on OpenVMS and other systems from multiple vendors, can be used to tie together clients and servers. NAS software integrates various client and server systems through application, communication, data interchange, and multivendor support (see Section 6.3). Complex information-sharing environments involving PC clients and operating system servers are supported by PATHWORKS software (see Section 2.2.2.2), which is part of NAS.

The following sections provide examples of different kinds of client/server relationships involving OpenVMS systems. Software in OpenVMS distributed production server environments is summarized in Section 2.3 and described in detail in Section 7.1. Software in client/server environments is described in Section 6.4.

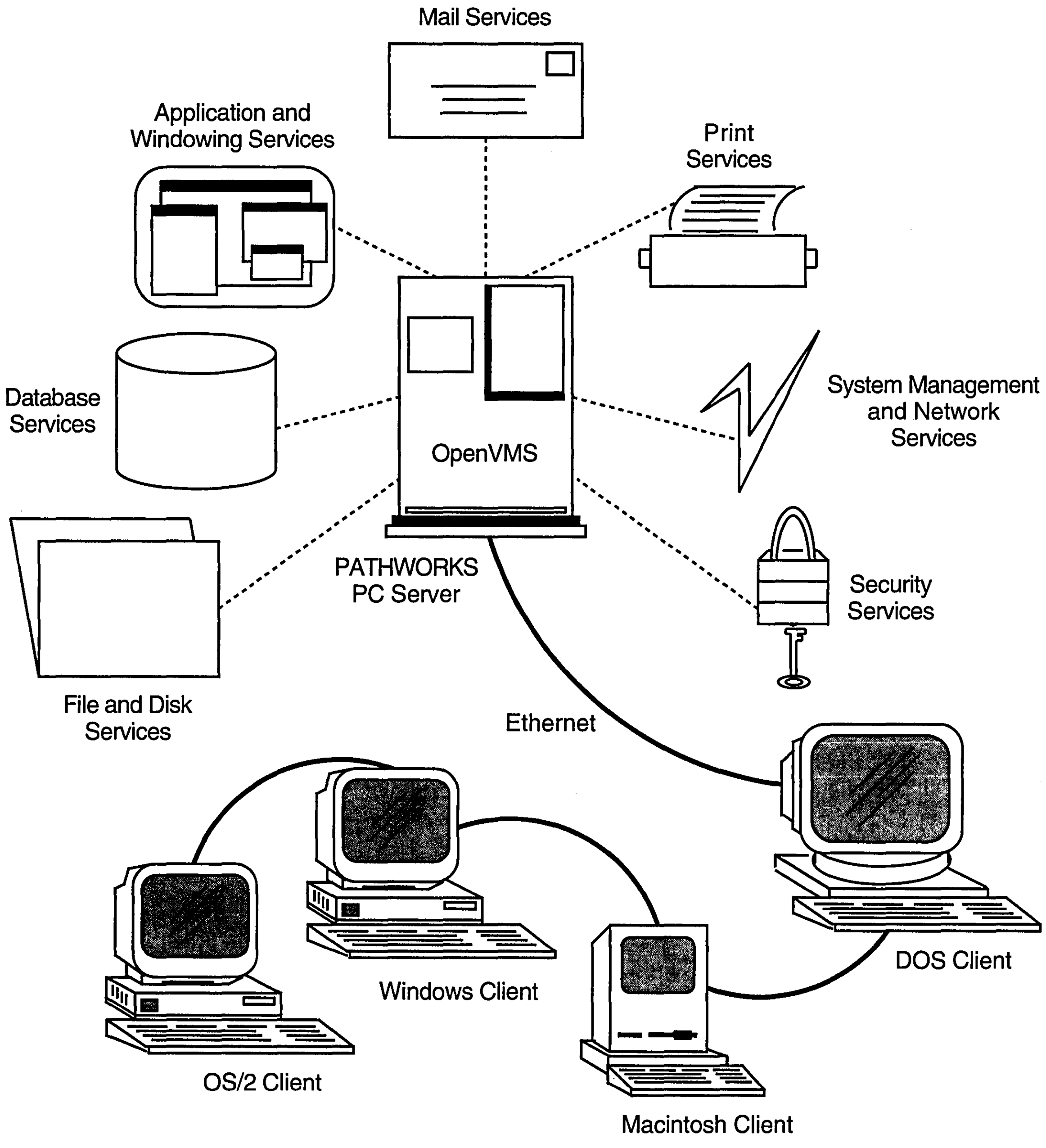
2.2.2.1 OpenVMS Servers with OpenVMS Clients

OpenVMS nodes in a VAXcluster are clients of distributed services provided by the VAXcluster system. A VAXcluster system can act as a disk, tape, file, print, and batch server to cluster members. VAXcluster systems can also provide database services to OpenVMS clients making SQL requests. Client/server relationships on VAXclusters are described in Section 6.4.1.

2.2.2.2 OpenVMS Servers with PC Clients

The OpenVMS operating system running on a VAX platform can act as an application, file, disk, and print server for large groups of personal computer clients through DECnet and/or TCP/IP networking connections. Servers and clients can be connected using PATHWORKS products: PATHWORKS server software on the OpenVMS or VAXcluster system, and PATHWORKS client software supported on a variety of personal computers (including MS–DOS, Windows, OS/2, and the Macintosh operating system), as illustrated in Figure 2–1.

Figure 2–1 OpenVMS Services to Personal Computer Clients in a PATHWORKS Configuration



ZK–5462A–GE

PATHWORKS is the personal computing system architecture that integrates personal, departmental, and enterprisewide computing environments so that PC users can directly access shareable information and resources throughout the enterprise. PC and terminal users can share applications, data, and system resources such as printers, disks, CD readers, and network gateways, without losing the benefits of industry-standard personal computing.

PATHWORKS for OpenVMS, when used in conjunction with DECnet for OpenVMS, enables the OpenVMS server to provide the following kinds of services (shown in Figure 2–1) to PC clients:

- File, disk, and print services
- Database services
- Electronic mail services
- Application and windowing services
- System management and network services
- Security services

The use of OpenVMS servers in PATHWORKS environments is described in detail in Section 6.4.2 and Section 6.4.3.

2.2.3 NAS Software in Distributed Environments

NAS software, based on open standards, provides services for distributed applications. The same services that run on OpenVMS can run on any distributed system that uses NAS products. NAS products provide the basic computing foundation to enable applications to be more interoperable and distributable across different computer systems: for example, OpenVMS systems, Digital operating systems based on UNIX (for example, ULTRIX), and other vendor systems (such as MS–DOS, OS/2, Macintosh, Sun, and IBM systems).

Using NAS software, applications can move easily across different systems in a network and have the systems all work together as if they came from one vendor. With applications based on NAS, users can more easily access and share information, no matter where they are located or what kind of system they are using.

2.2.4 Distributed Software Compliant with OSF Standards

The Open Software Foundation is specifying a Distributed Computing Environment (DCE), consisting of a set of base mechanisms that allow application developers to design distributed applications and systems that will interoperate. DCE capabilities extend system-level services to applications. OpenVMS support for DCE services is designed to allow OpenVMS applications to run on other systems that support DCE and permit applications designed for other systems to run on OpenVMS systems.

The following DCE technologies are included in the DCE Developer's Kit for OpenVMS:

- DCE Remote Procedure Call (RPC): The mechanism that provides transport and operating system independence; used in client/server applications. Includes a local directory service (LDS).
- DCE Threads Service (as supported by the DECthreads package described in Section 4.3): The mechanism for managing multiple threads of execution within a single process.

Another DCE capability being implemented on OpenVMS is the Cell Directory Service (CDS). OpenVMS implementation of DCE capabilities is discussed in Section 6.3.2.

DECnet/OSI networking software supports the Digital Distributed Time Service (DECdts), a method for coordinating and setting system time over the network, and the Digital Distributed Name Service (DECdns), which provides for networkwide naming of objects (as described in Section 2.2.5).

Another technology OSF is specifying is the Distributed Management Environment (DME), which identifies a common framework for managing networked systems. For a description of POLYCENTER management products used in this distributed management environment, see Section 2.4.3.

2.2.5 Distributed Networking Capabilities

OpenVMS networking support for distributed processing is provided by DECnet software and by the DEC TCP/IP Services for OpenVMS product (see Section 1.3.4). The use of networking software in multivendor distributed environments is discussed in Section 6.2.

The DECnet family of communication products (software and hardware) allows the OpenVMS operating system to participate in the DECnet network. DECnet/OSI for OpenVMS VAX is based on the OSI model. Users of DECnet/OSI for OpenVMS VAX can choose between OSI and DNA networking protocols, which can be running simultaneously. OSI protocols permit communication with other vendors' systems that support OSI. The DNA protocols are the traditional networking protocols that permit communication with other systems supporting compatible versions of the DNA protocols.

DECnet/OSI for OpenVMS VAX is an optional networking product that is compatible with the DECnet for OpenVMS product, a Phase IV product. Nodes running DECnet/OSI for OpenVMS and DECnet for OpenVMS can be configured to coexist on the same network.

Other DECnet networking software supplies services for distributed processing over the network, including DECdts and DECdns. DECdts supplies time-synchronization features essential to networks that are running distributed applications. DECdns provides distributed applications with a consistent, networkwide set of names called a namespace. This corporate namespace makes it possible for users and applications to refer to resources on the network (such as files, disks, and nodes) by using a single name, without having to know where the resource is located. The DECdns software uses a client/server model.

DEC TCP/IP Services for OpenVMS, an optional product, provides for TCP/IP connections and supplies NFS server capabilities. These services permit OpenVMS to become a full participant in TCP/IP networks. TCP/IP is the protocol used by UNIX systems on the Internet. The DEC TCP/IP Services for OpenVMS include a set of industry-standard communication protocols (TCP, IP, FTP, Telnet, and other protocols), Digital Remote Procedure Call (DECrpc) for OpenVMS, and NFS server software. NFS permits UNIX clients to access OpenVMS files and files compatible with UNIX stored on the OpenVMS system. (For additional information about the TCP/IP services supported, see Section 6.2.2.)

2.2.6 Distributed Features of DECwindows Motif

DECwindows Motif is an X Window System that permits users to access application programs running on other machines in the network as if the applications were running locally. With the DECwindows software, multiple device-independent applications can run simultaneously in various separate workstation windows. Applications function as clients and the DECwindows program that responds to the applications is the DECwindows server. (The DECwindows Motif user interface is described in Section 5.2.2.)

2.3 High-Integrity Production System Capabilities

OpenVMS offers commercial-strength computing capabilities to meet the needs of production systems. A production system is a computing system or configuration critical to the operation of an organization. OpenVMS supplies the computing power and dependability features needed to keep a mission-critical production system available as required, while ensuring the integrity of the data. OpenVMS systems and related optional software support predictable, stable computing environments.

The high-integrity, availability, manageability, and security features of OpenVMS have been proven in applications tested and enhanced over the years. Applications developed on OpenVMS systems cover the full range of activity from technical research to commercial datacenters.

The following sections discuss OpenVMS production system capabilities, summarize the characteristics of a dependable OpenVMS system, and describe OpenVMS software availability tools and data integrity tools. They also summarize database and transaction processing capabilities and manageability and security features for production systems. For a discussion of the growth potential of OpenVMS systems, see Section 1.5.

(Software used in the OpenVMS distributed production server environment is described in Section 7.1.)

2.3.1 Dependable OpenVMS Computing Systems

A dependable computing system is one that can be relied on to provide services to its users when the services are needed. A dependable system also provides sufficient performance so that users and applications can conduct their work efficiently.

The following types of OpenVMS systems meet different levels of dependability requirements for production systems:

- Conventional computing systems (for example, a standalone system or systems connected by a LAN).
- Highly available computing systems (for example, a VAXcluster system using shadowed disks and transaction processing software): For business functions requiring computing services that are uninterrupted for essential time periods, with minimal down time.
- Fault-tolerant computing systems (for example, the fault-tolerant VAXft processor): For business functions that require totally uninterrupted computing services (services critical to the business).

- Disaster-tolerant computing systems (for example, the VAXcluster Multi-Datacenter Facility [MDF] consisting of two linked datacenters located at a safe distance from each other): For business operations that must remain uninterrupted and must be immune to any type of foreseeable disaster.

Examples of components of a dependable system are VAXcluster systems, VAXft systems, transaction processing monitors, and database management systems.

The components in a dependable system have the following characteristics:

- Reliability: The ability to maintain a functioning condition, through the use of fault-prevention strategies.
- Recoverability: The ability to return to a functioning condition from a nonfunctioning or incorrectly functioning condition, through the use of failure-recovery strategies.
- Fault tolerance: The ability to exhibit an apparently functioning condition, through the use of error detection and correction strategies.

2.3.2 Availability Tools

OpenVMS availability tools include VAXcluster systems, volume shadowing software, and tools such as the DEC Availability Manager for Distributed Systems (DECamds):

- VAXcluster systems are loosely coupled multiprocessor configurations that allow designers to configure redundant hardware that can survive equipment failures. The OpenVMS operating system, which runs on each processor node in a VAXcluster system, provides a high level of transparent data sharing and independent failure characteristics. The processors interact to form a cooperating distributed operating system. All disks and their stored files are accessible from any processor as if those files were connected to a single processor. (See Section 1.3.3 for summary information on VAXcluster configurations and Section 3.4.1 for a description of the VAXcluster software.)
- Volume shadowing is a technique that provides data availability to computer systems by protecting against data loss from media deterioration, communication path failures, and controller or device failures. The process of volume shadowing entails maintaining multiple copies of the same data on two or more physical volumes. Up to three physical devices are bound together by the volume shadowing software into a shadow set or virtual unit. Volume shadowing software replicates data across the physical devices. Applications access the virtual unit as if it were a single standard, physical disk. If one volume of the shadow set fails, requests for data are automatically directed to another volume, allowing operation to continue without interruption.

  Phase II Volume Shadowing for OpenVMS is a fully distributed, clusterwide data availability product intended to service Digital and SCSI storage architectures. All essential shadowing functions are performed within the OpenVMS operating system (see Section 3.4.2).
- DECamds, which monitors, investigates, diagnoses, and corrects OpenVMS system resource utilization, permits the system manager to identify and resolve areas of resource denial on every system on the network. (DECamds is described in Section 7.1.2.1.)

2.3.3 Data Integrity Tools

Data integrity tools available on OpenVMS systems include RMS Journaling software and DECdtm software:

- RMS Journaling software helps protect the data integrity of Record Management Services (RMS) files on OpenVMS systems by recording file updates in a separate journal file. The journal file is used to restore the file to its original state if a failure should occur. Recovery-unit journaling ensures RMS operations on a file are either all completed or not done at all, to protect transaction integrity. (See Section 3.4.3 for a description of RMS Journaling software.)
- The Digital Distributed Transaction Manager (DECdtm) is a set of services to facilitate transaction processing. These services enable the application designer to implement atomic transactions, using an optimized two-phase commit protocol. DECdtm provides the enabling technology and features for distributed transaction processing, ensuring both transaction and database integrity across multiple resource managers (see Section 3.1.6).

2.3.4 Database and Transaction Processing Capabilities

OpenVMS supports optional software products that provide essential production system capabilities, as follows:

- Databases: DEC Rdb is a high-performance, high-capacity relational database for OpenVMS systems. It helps ensure data integrity and security for production systems. DEC DBMS is a high-performance CODASYL-compliant database management system, designed for high transaction volumes involving complex but stable relationships. Databases are described in Section 7.3.
- Transaction processing monitor: VAX ACMS (Application Control and Management System) is the highly reliable transaction processing monitor that works with other Digital commercial software to provide development and run-time environments for transaction processing applications. VAX ACMS provides dependability features, including application failover, to keep applications running reliably. VAX ACMS is suitable for very large online transaction processing (OLTP) applications for mission-critical business use. VAX ACMS is described in Section 7.2.2.
- The Digital forms management package, DECforms, integrates with VAX ACMS to provide forms processing capabilities in transaction processing environments.

In addition, the DEC Reliable Transaction Router (RTR) is a high-availability application used in several very large production systems around the world (for example, in stock/option markets). DEC RTR is described in Section 7.2.4.

A transaction may span multiple nodes of a cluster or network. Support provided by DECdtm services allows multiple resource managers, such as the DEC DBMS, DEC Rdb, and OpenVMS RMS software products, to be combined in a single transaction. Applications can define distributed transactions that may include calls to any of the Digital data management products. The distributed transaction will either commit or abort as a single transaction.

The OpenVMS system facilitates database sharing among applications that perform transaction processing and those that do not, and sharing with decision support systems and remote nodes requesting data.

2.3.5 Manageability and Security for Datacenters

OpenVMS provides a wide range of tools and utilities to help users manage complex computer environments, such as a large datacenter. Basic OpenVMS system management capabilities are integrated in the OpenVMS operating system and are augmented by optional products, such as the VAXcluster Console System, which serves as a control center for managing all connected VAX systems. Overall OpenVMS system and network management capabilities are described in Section 2.4 and optional management products running on an OpenVMS production server are summarized in Section 7.1.3.4.

- The optional VAXcluster Multi-Datacenter Facility (MDF) is designed to permit management of disaster-tolerant configurations. The MDF is an integrated software package that permits VAX CPUs in multiple datacenters to be combined into a single functional, security, and management domain. The MDF integrates clustering, volume shadowing, and related management and interconnect technologies into a manageable VAXcluster system, consisting of datacenters at widely separated locations connected by FDDI. The MDF can be managed from either location, even if a total datacenter failure occurs.
- Storage management software developed by the Digital Enterprise Storage Management (ESM) program provides software tools for managing mass storage and the data objects (files) stored on it in a distributed environment of heterogeneous computing systems. Examples of optional storage management products that run on OpenVMS are:
  - DEC File Optimizer for OpenVMS: Provides for file defragmentation
  - VAX Storage Library System (VAX SLS): Automates VAXcluster backup operations
  - VAX Disk Striping Driver for OpenVMS: Combines multiple disks into a single striped virtual disk (called a stripeset)

  These tools are described in Section 7.1.3.1.
- OpenVMS also supports sophisticated tools for monitoring and tuning performance. The capabilities of existing performance management and capability planning products have been incorporated into an integrated product set called DECperformance Solution (DECps) (as described in Section 7.1.3.3).
- The DECram for OpenVMS device driver, which creates a pseudodisk in main memory, can be used to improve I/O performance.
- The integrated security controls designed into the OpenVMS operating system (described in Section 3.3), combined with the capabilities offered by optional products that run on OpenVMS, guard data integrity and availability. In a local area network environment, optional encryption and decryption services provide protection for data being passed between systems.

2.4 System and Network Management Capabilities

OpenVMS reduces the complexity of managing individual systems, clusters of systems, and entire multivendor computing environments through the use of its own built-in management capabilities and optional software that runs on OpenVMS.

2.4.1 Managing OpenVMS Systems

OpenVMS provides, as an integral part of the operating system, system management capabilities that minimize the requirement for staffing of the system management function. Basic OpenVMS management capabilities include performance monitoring, tuning, security, storage management, and queue management (see Section 3.2). OpenVMS also supports a large number of applications and services provided by Digital and independent software developers for managing computer operations, including multivendor configurations.

OpenVMS system management tasks are ordinarily performed by a system manager. On a standalone VAXstation, the OpenVMS user usually serves as the manager of the OpenVMS system. In large installations, more than one system manager may be involved and some tasks may be performed by a computer operator.

OpenVMS systems to be managed range from standalone VAXstation workstations to large VAXcluster systems composed of VAX mainframe computers. Managing a VAXcluster system is similar to managing a single system. In general, the tools used and the security and performance considerations are the same.

Some integrated management utilities are applicable to all OpenVMS systems and clusters. Examples are the Backup, Monitor, and Mount utilities. The SYSMAN utility is used to perform cluster management tasks, such as setting up a clusterwide environment and executing commands on a cluster. (For a description of OpenVMS system management software, see Section 3.2.)

Optional software applications supply system management functions for complex VAXcluster environments. Examples are the DECscheduler, DECalert, and the VAXcluster Console Center. The optional VAXcluster Multi-Datacenter Facility (MDF) manages large disaster-tolerant configurations.

OpenVMS can manage other systems remotely over the network. The VAX Remote System Manager (VAXrsm), which runs on OpenVMS, is an optional networking product that permits a system manager to manage a number of OpenVMS systems and ULTRIX systems connected to a DECnet network.

Using an OpenVMS management server can simplify management of remote systems and software. In a complex networking environment, a management server can provide effective control of distributed resources.

2.4.2 Managing Networks

Digital integrates network management operations into the network itself to keep applications and enterprises running despite changes in the network. The network automatically collects and stores data about itself, reports on network events, and takes action based on this information. Users can access this information and make online adjustments from anywhere in the network using network management software that exists on each node in the network.

DECnet/OSI provides modular, distributed network management capabilities, permitting remote management of all network functions. Network management is defined in the framework of the Digital Enterprise Management Architecture (EMA), as described in Section 2.4.3. The command line interface to DECnet/OSI network management is the Network Command Language (NCL), which is used to manage all DECnet/OSI nodes and their entities.

The traditional DECnet for OpenVMS software employs Network Control Program (NCP) commands to interface with Phase IV network management. DECnet/OSI and DECnet Phase IV are compatible. DECnet/OSI nodes can invoke NCP to manage existing Phase IV nodes on the same network.

For additional information about DECnet network management in an open, distributed environment, see Section 6.2.

2.4.3 Managing Integrated Enterprises

OpenVMS is providing support for the standards-based Enterprise Management Architecture, which is intended to manage complex, distributed, multivendor computing environments. EMA components include the manageable units called entities and the directors that manage the entities.

EMA is the foundation for the Digital POLYCENTER management products for integrated management of networks, systems, applications, and databases from many sources. POLYCENTER applications are designed to interoperate with applications and frameworks that comply with the OSF Distributed Management Environment (DME).

POLYCENTER products include the DECmcc family of products. DECmcc is a multiplatform software system that can be extended to manage practically any entity across an enterprise environment. The DECmcc Director, supplied to OpenVMS customers with DECnet/OSI for OpenVMS VAX, provides an open, extensible management platform for integrating network and system management applications.

The use of EMA-compliant software to manage enterprisewide multivendor environments is described in Section 6.3.4.

Part II

OpenVMS Systems Software

This part of the manual provides a summary overview of the software capabilities of the OpenVMS system and related optional software products, as used in an OpenVMS standalone or VAXcluster configuration.

Chapter 3 describes the base OpenVMS operating system components and optional software integrated in the operating system. Chapter 4 summarizes programming software integrated in the system that supports application program development. Chapter 5 describes user interfaces and general user software provided with the OpenVMS system.

As described in this part of the manual, standalone OpenVMS systems and OpenVMS systems in a VAXcluster configuration can supply full capabilities for interactive, timesharing, and real-time processing. For a description of the use of OpenVMS and VAXcluster software in distributed, multivendor computing environments, see Part III.

3 Description of OpenVMS System Software

The OpenVMS operating system is the group of software programs that control computing operations and coordinate and manage the resources of the processing system. The software is loaded into memory when the system is initialized (booted) and remains there until the system is shut down. It performs both automatically and manually and supplies the operating environment for the user.

This chapter presents a summary functional description of the software components of the base OpenVMS operating system, including system management and security software. It also describes optional software integrated into the OpenVMS system, including VAXclusters, volume shadowing, and RMS Journaling software.

3.1 OpenVMS Operating System Components

The OpenVMS operating system is made up of thousands of program modules that are grouped into layers (as shown in Figure 3–1):

- OpenVMS operating system kernel, supported by system services
- OpenVMS core services
- OpenVMS utility programs

In the OpenVMS system, each program (known as an image) executes in the context of a process, the basic entity scheduled by the OpenVMS system software. A process consists of a process address space and hardware and software context.

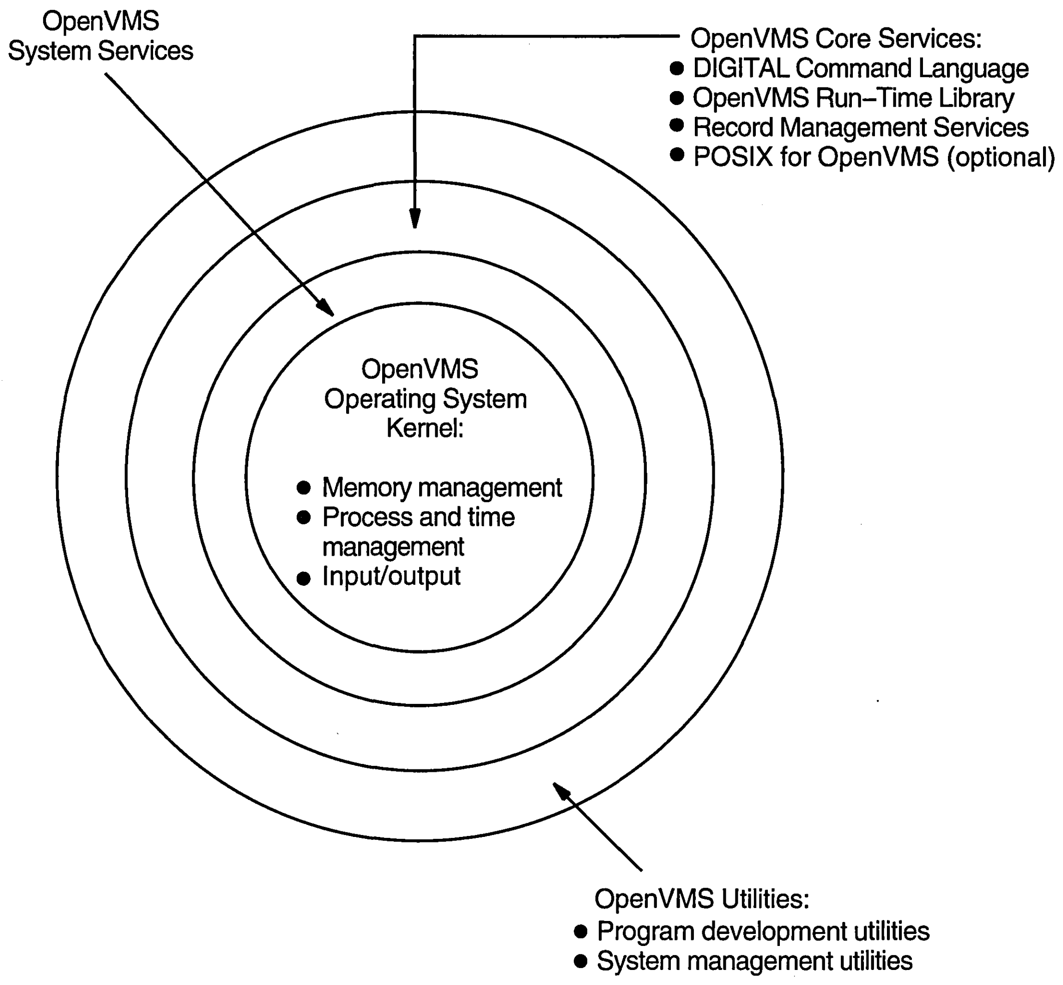
3.1.1 OpenVMS Kernel

The OpenVMS kernel includes the programs, data structures, and related system services essential to the operating system. The programs in the kernel perform and coordinate processor operations and provide a multiuser, multitasking operating environment. The kernel programs are composed of three subsystems:

- Memory management subsystem
- Process and time management subsystem
- Input/output (I/O) subsystem

The memory management subsystem controls the allocation of memory resources and implements the OpenVMS system's virtual memory operation. Virtual memory management permits the development and use of programs larger than physical memory and allows any number of jobs to share memory. To optimize the use of physical memory, the memory management subsystem causes large segments of code and data to be moved in and out of physical memory as they are needed. This process is carried out by the swapper mechanism. The basic set of memory locations used in memory mapping is called the page. (The value of

Figure 3–1 OpenVMS Operating System Components


OpenVMS System Services
OpenVMS Core Services:
● DIGITAL Command Language
● OpenVMS Run–Time Library
● Record Management Services
● POSIX for OpenVMS (optional)
OpenVMS Operating System Kernel:
● Memory management
● Process and time management
● Input/output
OpenVMS Utilities:
● Program development utilities
● System management utilities

ZK–5484A–GE

the page is CPU specific.) The paging mechanism brings pages of an executing process into physical memory when referenced. When a process is executing, all of its pages are said to reside in virtual memory.

Process and time management functions include scheduling jobs for processing and performing process control. OpenVMS schedules processor time and memory residency on a priority basis. All processor operations are assigned a priority and executed based on that priority. When multiple requests are waiting at the same time, the scheduler starts and stops jobs on the basis of their priority. When an event (such as an I/O interrupt) occurs, the system processes the event and passes control to the highest priority process ready to execute.

The OpenVMS I/O subsystem consists of OpenVMS device drivers and their associated data structures, as well as related system services. The I/O subsystem is responsible for processing I/O requests received from other layers of the OpenVMS operating system or from user-application programs. Device drivers execute I/O instructions to transfer data to and from the device. Each type of I/O device requires its own driver. Digital supplies drivers for all devices supported by the OpenVMS operating system and provides $QIO service routines to access the special device-dependent features available in many of these devices. Users can also write their own device drivers if they have special needs or are using devices not supported by OpenVMS. (The types of I/O connections possible on OpenVMS systems are listed in Section 1.3.1.)

Systemwide, protected data structures that support the kernel software include pager data structures, the I/O database, and scheduling queues.

The OpenVMS system services support and complement the kernel subsystems. The system services control system resources available to user applications, provide communication and synchronization among processes, and coordinate I/O operations. User-written application programs can call system services directly (see Section 4.3.2).

OpenVMS provides the following interprocess communication facilities for applications that consist of multiple cooperating processes:

- Shared memory sections that permit multiple processes to have concurrent access to shared address space
- Common event flags that provide simple synchronization
- Mailbox virtual devices that allow processes to communicate with queued messages
- The lock manager, which provides a facility for synchronizing access to resources by allowing locking and unlocking of resources by name (see Section 3.4.1)

OpenVMS POSIX (the Portable Operating System Interface) supports real-time functions for interprocess communication between multiple processes in the area of event notification, message queues, and shared memory (see Section 4.4). POSIX applications developed in the OpenVMS POSIX environment can call on OpenVMS services directly. Use of system services that are specific to the OpenVMS operating system, however, will affect the portability of the POSIX application.

3.1.2 OpenVMS Core Services

OpenVMS core services support basic user and application interaction with the processing system. The core services include the following:

- DIGITAL Command Language (DCL): Command-line interface to the OpenVMS operating system
- OpenVMS Run-Time Library: A library of run-time routines for performing a variety of commonly required functions
- OpenVMS Record Management Services (RMS): A device-independent method of handling I/O
- Optionally installable POSIX for OpenVMS: Portable operating system interfaces that conform to IEEE standards and enable users to write applications that can be moved from one system conforming to POSIX and X/Open standards and draft standards to another and used on both of them

DCL is a standard user interface to OpenVMS that supports batch or interactive operations. DCL provides commands for requesting system actions, from manipulating files to running applications (see Section 5.2.1).

The OpenVMS Run-Time Library provides commonly required application functions and other general-purpose functions. These routines can be called by programs written in all the programming languages supported by OpenVMS. All routines in the run-time library follow the OpenVMS calling standard and most are contained within a shareable image. (OpenVMS Run-Time Library routines are described in Section 4.3.1.)

The OpenVMS RMS facility provides a set of general routines for file and record operations on OpenVMS systems. RMS is a high-level interface to the file system and OpenVMS I/O subsystem. It is used by most optional (layered) products for file and record operations and is the default I/O service for all programming languages that run on the OpenVMS system. RMS routines can be used by application programs for creating files, opening and closing files, and reading, writing, and deleting records in files. (RMS is described in Section 4.8.)

POSIX for OpenVMS (described in Section 2.1.3.1) is optional software which, when installed, is tightly integrated into the OpenVMS environment and can take advantage of OpenVMS features. However, if an OpenVMS POSIX application uses features available in the OpenVMS environment that do not conform to POSIX and X/Open standards and draft standards, the portability of the POSIX application will be affected. OpenVMS technologies, such as clustering and volume shadowing, are available transparently to POSIX applications. (OpenVMS POSIX programming is described in Section 4.4 and the OpenVMS POSIX user interface in Section 5.3.)

3.1.3 OpenVMS Utility Programs

The OpenVMS utility programs include system management and user utilities and program development facilities. See Table 3–1 for examples of OpenVMS utility programs.

Table 3–1 Examples of OpenVMS System Utility Programs

Type of Utility	Examples of OpenVMS Utility Programs
System management utilities	Authorize, AUTOGEN, Backup, Monitor, Mount, SYSMAN
User utilities	Mail, Sort/Merge, Help
Text processing utilities	Digital Standard Editor (EDT), Extensible VAX Editor (EVE)
Program development utilities	Linker, Debugger, Librarian, File Definition Language (FDL)

OpenVMS provides system managers and users with many powerful utilities to control the day-by-day operations of their systems. OpenVMS system management utilities are described in Section 3.2. OpenVMS users can access utilities that assist them in performing daily operations and communicating with other users. User utilities provided with the OpenVMS operating system, including editors and text processors, are discussed in Chapter 5. OpenVMS programming utilities include text processors for creating source programs and program development utilities for processing the programs. These programming utilities are described in Chapter 4.

Other optional software that runs in OpenVMS environments is described in Chapter 6 and Chapter 7.

3.1.4 OpenVMS Vector Processing Capability

The OpenVMS operating system provides fully shared, multiprogramming support for VAX vector processing systems. A vector is a group of related scalar values or elements (such as an array of numbers). Several midrange and high-end VAX processors have adopted an optional design for integrated vector processing using vector registers and vector instructions. A vector processor can routinely process a vector four or five times faster than a traditional processor. Vectorization is useful for CPU-intensive applications involving repeated operations on groups of simple elements (for example, in weather forecasting, molecular modeling, and financial modeling).

OpenVMS support for vector processing includes these features:

- Fast-vector math routines for high performance
- A standard OpenVMS facility to permit writing and debugging of vectorized applications on an OpenVMS system that emulates the vector processing environment
- Vector processing operations supported by system management and the OpenVMS Accounting and Error Log utilities

3.1.5 OpenVMS Symmetric Multiprocessing

OpenVMS supports symmetric multiprocessing (SMP), which permits multiple processors in an SMP configuration to perform operations simultaneously, dividing and sharing the work load. SMP multiprocessors are tightly coupled and appear to the system and the user as a single system (see Section 1.2.2). OpenVMS SMP is available on selected VAX processing systems.

Multiple processors execute code from a single shared memory address space, and users and processes share a single copy of OpenVMS. SMP also provides simultaneous shared access to common data in global sections available to all processors. SMP dynamically balances the execution of all processes across all processors, based on process priority.

The primary advantage of SMP is increased throughput. SMP multiprocessing can be optimized in two ways:

- Multiple processes can be executed simultaneously on different processors, maximizing overall system performance.
- Single-stream application programs can be partitioned into multistream jobs, minimizing the processing time for a particular program.

3.1.6 Digital Distributed Transaction Manager

The OpenVMS operating system includes Digital Distributed Transaction Manager (DECdtm) services, which provide the basic operating system support for distributed transactions. An individual transaction is a series of operations on a file.

DECdtm services comprise a transaction manager, interfaces to system services, communication services, and logging and recovery services. The transaction manager supports application-supplied services to start, end, or abort transactions, and sends instructions to resource managers on how to complete the transaction.

DECdtm integrated services for distributed transaction management ensure both transaction and database integrity across multiple resource managers (such as database systems) and across networks around the world. To ensure consistency of data, DECdtm services cause updates to all databases to occur as a single "all or nothing" unit of work, regardless of where the data resides physically.

DECdtm employs a two-phase commit protocol that enables application developers to build dependable distributed applications. The two-phase commit protocol is a handshaking procedure in which all participants in a distributed transaction first agree to commit and then, on a signal from coordinating software on the node that initiated the transaction, actually do commit. The commitment protocol guarantees that all parts of a transaction complete or the transaction has no effect on any involved resource manager (see Section 7.2.3).

DECdtm services permit applications to define distributed transactions that can include calls to Digital data management products. Regardless of the mix of data management products used, the distributed transaction will either commit or abort. DECdtm system services are supported by an array of products that run on OpenVMS, including transaction processing monitors, database products, and RMS Journaling. Transaction processing and database products are described in Chapter 7.

3.2 OpenVMS System Management Software

OpenVMS system management tasks ordinarily include the following:

- Installing, upgrading, and updating software
- Starting up and shutting down the system
- Customizing the system
- Managing user access to the system and controlling system resources
- Managing and monitoring system security
- Managing peripheral devices and storage media, including public disks
- Managing system files and directories
- Backing up and restoring files
- Monitoring the system
- Tracking and reporting resource usage
- Maintaining acceptable system performance and tuning the system
- Managing batch and print queues

System managers may also be called upon to perform some of these tasks:

- Setting up and maintaining a network
- Setting up a VAXcluster environment
- Managing special system configurations or processing environments

3.2.1 OpenVMS Installation and Configuration

OpenVMS supplies an interactive VMSINSTAL command procedure that guides the system manager through each step of installing and updating OpenVMS operating system software and upgrading OpenVMS software to a higher version number. The same command procedure permits the system manager to install and upgrade optional (layered) products on the OpenVMS system.

During the OpenVMS installation procedure, VMSINSTAL sets up initial system parameters and installed images using AUTOGEN software and automatically brings up the system.

Starting up a newly installed OpenVMS system involves booting the system, using either a nonstop or conversational boot. Nonstop booting is faster and easier because Digital sets typical system parameters for the hardware configuration. Conversational booting permits the system manager to change SYSGEN (System Generation) utility parameters during the boot procedure. On a standalone system, to reduce startup time, a system snapshot can be taken and used for booting the system.

OpenVMS will boot on the VAX processor, automatically configuring itself to the CPU, memory, and devices present. After a new configuration is booted, the system manager can use the AUTOGEN command procedure with its feedback mechanism to optimize system parameter settings.

When the system starts up, the system manager can log in and set up the environment. Each time the system starts up, it executes a series of startup command procedures, some of which the system manager can edit to customize the user environment. The startup command procedure invokes the SYSGEN utility to poll all connected I/O buses for devices and then configures itself to allow access to those devices.

Examples of some OpenVMS utilities and DCL commands that pertain to system configuration tasks are given in Table 3–2.

Table 3–2 Examples of OpenVMS Configuration Utilities and Commands

System Management Task	Utility or DCL Command
Automatically set system parameters by detecting devices installed in a configuration	AUTOGEN command procedure
Create an on-media file system	INITIALIZE command
Make a volume known to the system	MOUNT command
Manipulate device characteristics	SET DEVICE command
Start or stop CPUs in an SMP kernel	SET CPU command
Manipulate magnetic tape characteristics	SET MAGTAPE command
Manipulate printer characteristics	SET PRINTER command
Manipulate terminal characteristics	SET TERMINAL command
Load and connect device drivers	SYSGEN utility
Define the system management environment across nodes in a VAXcluster; enable device and processor control commands to take effect across a cluster	SYSMAN utility

(continued on next page)

Table 3–2 (Cont.) Examples of OpenVMS Configuration Utilities and Commands

System Management Task	Utility or DCL Command
Manage startup file configuration	SYSMAN STARTUP command
Manage disk space quotas	SYSMAN DISKQUOTA command
Automate the installation of software updates to OpenVMS and optional software	VMSINSTAL command procedure

3.2.2 Management Software for the General OpenVMS Environment

In addition to configuring the OpenVMS system environment, the OpenVMS system manager performs a variety of tasks to manage the environment. Examples of OpenVMS utilities used to perform general system management tasks are given in Table 3–3.

Table 3–3 Examples of OpenVMS System Management Utilities

System Management Task	Utility
Report on system use and account for use of system resources	Accounting utility
Control access to the system and its resources; use commands to set up user accounts (SYSUAF.DAT)	Authorize utility
Perform backup of full volume or incremental files on mounted disks to tape or another disk and restore them	Backup utility
Selectively report the contents of the error log file containing system error messages	Error Log utility
Install shared images on the system	Install utility
Set up an OpenVMS system to be a service node on a LAT configuration	Local Area Transport Control Program (LATCP)
Manage multiple software licenses	License Management Facility (LMF)
Monitor system activity	Monitor utility
Make the contents of a tape or disk available to system	Mount utility

The system manager is responsible for managing user access to the system. The Authorize utility permits the system manager to establish the individual controls that customize user accounts. When a user logs in to OpenVMS, the system uses the information in the user authorization file (UAF) to validate the login attempt, establish the account environment, and create a process with the specified attributes. The system manager can use the Authorize utility to add to or modify user records in the UAF and other security-related system databases. (Section 3.3 describes security management in the OpenVMS system environment.)

Management of the OpenVMS environment includes management of peripheral devices and storage media. Peripheral devices include storage devices (such as disks, compact discs, and tapes) and input/output devices (such as terminals, terminal servers, printers, modems, and card readers). System managers or operators can use DCL commands to add devices, set device software characteristics, display device information, and set protection for the devices.

Storage management is increasingly important as the amount of data to be stored grows. Very large configurations supporting thousands of users generate a huge volume of data. The Digital approach to storage management is based on the foundation of OpenVMS computing and includes optional network-based storage management products.

OpenVMS provides the tools to allocate, initialize, and mount storage devices, create disk volume sets, and set protection on volumes. The system manager can create logical names for files and directories and can plan and create public directories. The DCL command ANALYZE/DISK_STRUCTURE can be used to check and repair disk structure. Defragmentation of disks can be performed using the OpenVMS Backup utility or the optional DEC File Optimizer for OpenVMS. (Optional storage management products developed by the Digital ESM Program are described in Section 7.1.3.1.)

One of the system manager's responsibilities is to prevent the loss or destruction of data due to equipment failure or accidental deletion or corruption of files. The system manager or operator can use the OpenVMS Backup utility to make backup copies of volumes on magnetic tape, magnetic disk, or certain optical disks. The Backup utility provides full volume and incremental file backup for file-structured, mounted volumes and volume sets. Individual files, selected directory structures, or all files on a volume set can be backed up and restored.

In addition to routine, regular backup of files, the system manager can perform a standalone backup to back up and restore the system disk. A standalone backup can also be done during the installation of the operating system. The Backup utility can be used to restore a save set (a special file created by the Backup utility) or list the contents of a save set.

The system manager can also assign a protection mask to files, devices, and other objects to prevent unauthorized access (see Section 3.3.2).

The system manager is responsible for ensuring that the system performs consistently at an acceptable level. OpenVMS provides software tools that help the manager to monitor the condition of the system and to maintain and tune the system.

Monitoring the system involves obtaining information about the system, its users, and the processes running on the system. Examples of software used to monitor the system include the following:

- DCL SHOW commands display information about system, device, and process conditions (providing options such as CLUSTER, DEVICES, ERROR, MEMORY, STATUS, SYSTEM). DCL command procedures can be developed to automate monitoring of system information.
- The Monitor utility is used to monitor different classes of systemwide performance data and to display it or save it in a file for later use. For example, the following can be monitored: process activity, I/O activity, memory management activity, vector processing activity, and two-phase commit transaction activity at specified intervals.
- The Accounting utility is used to monitor resource usage. The utility processes system accounting files to produce reports and summaries that indicate the ways the system is used, how it performs, and, in some cases, how particular users use the system.
- Log files that report on system activity include the following:
  - Operator log file: The operator can access this file directly.

- Error Log file: The system manager or operator can use the ANALYZE/ERROR_LOG command to analyze and view data in this file.
- Security audit log file: The manager can use the Audit Analysis utility to analyze this file.

Maintaining acceptable system performance involves using the Monitor utility to develop a database of performance information and using the Accounting utility to generate performance reports. With this information, the system manager can manage the work load on the system. For example, the manager can distribute work to off hours (using batch queues) and limit the number of concurrent interactive users (using DCL commands or Authorize utility commands).

The AUTOGEN command procedure, run at system installation, sets a number of system parameters by detecting devices installed in a configuration. AUTOGEN can also be run on a regular basis using a command procedure. The AUTOGEN feedback option generates a report of recommended parameter settings for system tuning. The SYSMAN utility can also be used to set system parameter values.

OpenVMS can adjust itself dynamically during operation. Built-in adjustment capabilities optimize the OpenVMS system to meet the needs of particular environments (such as timesharing, transaction processing, batch processing, or real-time processing) and different types of application work load (such as business or engineering).

In addition to performance capabilities provided by OpenVMS operating system software, the system manager can optionally employ other performance management software, particularly useful in large, complex environments such as production system environments. Optional performance management software suited for use in the OpenVMS production server environment is described in Section 7.1.3.3.

OpenVMS facilities to perform batch and print operations involve the creation of queues and the setup of spooled devices to allow processing of batch, timesharing, and real-time jobs. The OpenVMS operating system provides generic queues that hold batch or print jobs until appropriate execution queues become available.

The system manager can use DCL commands to regulate the number of queues and the number of batch jobs in the queue that can execute concurrently. The manager can set queue attributes, start and stop queues and the jobs in queues, and perform other queue maintenance operations, as illustrated by the examples in Table 3–4.

Table 3–4 Examples of OpenVMS Queue Management Commands

System Management Task	DCL Command
Delete a print or batch queue and all the jobs in the queue	DELETE/QUEUE command
Create or initialize queues and assign names and attributes to the queues	INITIALIZE/QUEUE command
Start or restart the specified queue after it has been initialized	START/QUEUE command

(continued on next page)

Table 3–4 (Cont.) Examples of OpenVMS Queue Management Commands

System Management Task	DCL Command
Pause the specified queue and suspend all the jobs currently executing on the queue	STOP/QUEUE command
Stop all queues on a node	STOP/QUEUE/ON_NODE command
Change the attributes of the queue	SET QUEUE command
Display information about queues and the jobs currently in queues	SHOW QUEUE command
Display information about a user's batch and print jobs or about specific job entries	SHOW ENTRY command
Queue one or more batch jobs to a batch queue	SUBMIT command
Queue one or more files for printing to an output queue	PRINT command

3.2.3 Management Software for Specific Environments

A system manager can configure the OpenVMS system as a node in a network and can establish a VAXcluster system configuration. In addition, a manager can set up and manage special configurations and special processing environments. The optional OpenVMS POSIX environment also entails the use of installation and management techniques specific to OpenVMS POSIX.

The procedure for setting up the OpenVMS system as a network node depends on the networking software to be used. Networking software that runs on the OpenVMS system includes DECnet for OpenVMS (DECnet Phase IV) and the separately installable option, DECnet/OSI for OpenVMS VAX. The command language interface for DECnet Phase IV is the Network Control Program (NCP); for DECnet/OSI, it is the Network Command Language (NCL).

To prepare the OpenVMS system for the network environment, the system manager selects a node name and network address and makes the necessary hardware connections to the network. The manager performs network installation and configuration procedures, using the appropriate networking software tools, and starts the network software running on the system. To keep the network software running, the system manager uses network tools such as counters and event logging reports to perform basic problem solving for the local node. Overall network management is summarized in Section 2.4.2 and network management tasks and tools are described in Section 6.2.1. Refer to the appropriate networking product documentation for specific information on installing, configuring, and managing networking software.

To establish a VAXcluster system, the system manager prepares the cluster operating environment, sets up cluster disks and tapes and cluster queues, and builds the cluster. The interactive CLUSTER_CONFIG command procedure permits the manager to configure the cluster easily and directly. The command procedure can add or remove a computer from the cluster, change a computer's characteristics, or create a duplicate system disk.

Managing the VAXcluster system is similar to managing an individual OpenVMS system. The same tools and DCL commands are used to perform management tasks, with the exception that the SYSMAN (System Management) utility is used to perform clusterwide management. The SYSMAN utility provides the ability to centralize the management of a VAXcluster system by defining and then managing the entire cluster from a single node. The system manager defines a system management environment in which operations performed from the local VAX system can be executed on all other VAX systems in the defined environment. SYSMAN accepts the following sets of commands as well as most DCL commands: CONFIGURATION, STARTUP, PARAMETERS, and DISKQUOTA. The dynamic Show Cluster utility displays the status of VAXcluster hardware components and communication links. Managing security in the VAXcluster environment is described in Section 3.3.1.

Other possible management responsibilities for the OpenVMS system manager may include:

- Managing special system configurations (such as terminal and Infoserver connections). The LAT Control Program (LATCP) utility is used to configure and control the LAT protocol on OpenVMS host systems; LAT supports communication with terminal servers. The LAD Control Program (LADCP) configures and controls the LAD protocol for OpenVMS host systems. The LAD protocol permits a user to access InfoServer discs (CD–ROMs) and other disks and tapes as though they were locally connected to the OpenVMS system; thus, several OpenVMS nodes can share the same disk media.
- Managing special processing environments (such as transaction processing, symmetric multiprocessing, or vector processing environments). For example, the system manager can use the Log Manager Control Program (LMCP) utility to create and manage log files that are used by transaction managers (see Section 7.2.3) and can use the Monitor utility to monitor the status of transactions executing on the system.
- Installing and managing the OpenVMS POSIX environment. Refer to the *POSIX for OpenVMS Installation and System Management Guide* for detailed information about installing and managing OpenVMS POSIX on an OpenVMS system.

3.3 OpenVMS System Security

Safeguarding the information processed and stored on the OpenVMS system can be an important consideration, especially as many system environments become more open and distributed. OpenVMS provides an extensive range of built-in features that can be used to secure the OpenVMS computing environment, ensuring that information is available only to those who need it and that unauthorized access is prevented.

The following sections describe the security environment and security standards supported by OpenVMS. They identify the primary security features and security management software and tools.

3.3.1 OpenVMS Security Environment and Standards

Security concerns apply to whole environments, such as offices, because information processing no longer necessarily occurs only in limited, centralized datacenters. OpenVMS security features are designed to protect systems and information in any configuration or environment and apply to VAXcluster systems as well as single standalone OpenVMS systems.

OpenVMS offers integrated security capabilities that meet the C2 level of security defined in the Department of Defense System Trusted Computer System Evaluation Criteria (TCSEC), published by the National Computer Security Center (NCSC). The C2 criteria are the security capabilities required by government and defense contractors and also by many commercial customers.

The security requirements for a class C2 system are built into the OpenVMS operating system and include extensive audit trail capabilities and discretionary access controls.

Additionally, optional OpenVMS Security Enhancement Services (SEVMS) software provides class B1 functionality, which includes mandatory access controls, labeled object protection, and additional auditing.

In terms of security, the system is the computing and communication environment over which the manager has some control. The system protects everything inside the system. The subset of OpenVMS that is secure is called the Trusted Computing Base. It includes the executive and file system, other components that do not execute in user mode, system programs, and related system management utilities. The security domain controls access mediation, resource allocation, authorization, and the auditing subsystem.

The OpenVMS system may be connected to other systems in a network. The DECnet network software used to link the systems provides built-in security controls (for example, access controls for logical link connections). Optional encryption services protect messages transmitted between systems over the local area network (LAN) (as described in Section 6.2.3).

3.3.2 OpenVMS Security Management Software

The OpenVMS operating system provides built-in security features that can be implemented in a flexible manner. File protection can be extended to protect all, none, or some of the files. Passwords can be general, screened, locked, or eliminated. The auditing subsystem can be used to warn a system manager, operator, or security officer of attempted system break-ins or unauthorized access to system files or other objects.

Selective use of security features permits establishment of a high degree of security for sensitive data, while minimizing or eliminating control procedures for less essential data. Table 3–5 summarizes some significant security features provided by OpenVMS.

Table 3–5 OpenVMS Security Features

OpenVMS Security Feature	Values or Description
Login classes	Local, Remote, Dialup, Network, Batch, Process
Access modes	Interactive and Noninteractive
Password authentication and management	Security controls involving password length, password generation, system passwords, password dictionaries, password histories
User accounts as defined in the UAF	Regular, "captive," and restricted accounts
Break-in detection and evasion	Mechanisms for detecting break-in attempts and features for automatic evasion of break-in
Categories of file protection	System, Owner, Group, World
Rights for each category of file protection	Access rights: Read, Write, Execute, Delete, Control

(continued on next page)

Table 3–5 (Cont.) OpenVMS Security Features

OpenVMS Security Feature	Values or Description
Privileges	Granted to users according to their need to access system functions
Protection based on user identification code (UIC)	Determined by owner UIC and a protection code; controls access to objects (for example, files, directories, volumes, queues)
Access control list (ACL) protection	Matches specific access to specific users or groups of users for each object
Proxy access definition	Permits a user from a remote node to log in to a local node as if the user owned an account on the local node
Auditing and logging	Comprehensive auditing subsystem
Secure terminal path	Capability designed to thwart "password grabbers"
Highwater marking	Technique for discouraging disk scavenging

The OpenVMS system provides mechanisms for controlling user access, securing data and resources, and creating an audit trail.

The OpenVMS system manager can use OpenVMS security tools to define levels of protection for and control access to memory, files, devices, and other OpenVMS security objects. The manager can establish a security database that includes the following controls:

- For security subjects (user processes, jobs, and applications that need to access security objects), the manager sets up the system authorization file (SYSUAF), the user rights database file (RIGHTSLIST), and the network user authorization file (NETPROXY.DAT).
- For security objects, the manager specifies protection codes, the UIC assigned to the owner of the object, and the access control list (ACL) that defines the access granted to specific users. (Examples of security objects are files, devices, volumes, queues, global sections, logical name tables, common event flag clusters, resource domains, and capabilities.)
- The manager establishes an audit trail, a listing of enabled security-related events, that can be used to monitor system activity.

Within the security domain, authentication is the procedure for verifying a person's identity, through passwords. Mechanisms for identification on an OpenVMS system include login to a user account and proxy login. The OpenVMS system manager uses the Authorize utility to establish user accounts and controls passwords in the SYSUAF (see Section 3.2.2).

The Authorize utility permits the system manager to set up or modify an account that matches the individual user's needs. An interactive user who performs general work on the system needs an individual account and a file directory. Other users may require only a limited-access account to a restricted system environment, for example, to perform routine tasks (such as shop-floor operations), run batch operations during unsupervised periods, or run an application program containing private information. Turnkey or "captive" accounts limit the activities of the user and deny the user access to the DCL command level. Restricted accounts have features similar to those of a captive account but permit access to the DCL command level after login command procedures are executed (for example, to permit a user to access electronic mail).

Some of the information in the UAF account record is as follows:

- User name and password for the account (the password is encrypted)
- User identification code (UIC), which identifies the user as a member of a group that can share data
- The disk device and directory containing files owned by the account
- Limits and quotas on reusable system resources
- System privileges that define processing activities the account's process can engage in
- Any limitation on allowed login times
- Qualifiers such as "captive" or "restricted" that limit access to the system

OpenVMS supports discretionary access controls that permit individually named users to be either included or excluded for accessing a certain file or achieving particular forms of access.

In addition, the system manager can set up a certain application as a protected subsystem. In the protected subsystem, data files and resources are controlled by the subsystem and data can be accessed only by running the subsystem. When it is run, the subsystem causes the process running the application to be granted additional identifiers. The system manager grants the identifiers to the people who will serve as managers of the subsystems.

Security auditing facilitates tracking of sensitive system objects. Auditing enables the manager to monitor system activities and prevent unauthorized access. The OpenVMS audit trail mechanism allows users and security managers to record security-related events on an audit log file and possibly send them to an operator terminal, which may be designated as a security alarm console. The system manager can set security alarms for events like login failures, authorization changes, and file access. The manager or security administrator can use the Audit Analysis utility to extract audit trail information from the audit log file and can use the SET AUDIT command to select events to be audited.

3.4 Optional OpenVMS System Integrated Software

The following optional software capabilities are integrated into the OpenVMS operating system:

- VAXcluster system software
- Volume shadowing software
- RMS Journaling software

The following sections summarize the functions supplied by these OpenVMS integrated software products.

3.4.1 VAXcluster Software

The VAXcluster system provides a highly integrated OpenVMS computing environment distributed over multiple VAX CPUs. The individual CPUs that are members of a VAXcluster system can be connected by supported interconnects (as described in Section 1.3.3).

OpenVMS system managers can tailor the VAXcluster operating environment to create a common environment (with the same resources available on all members) or a multiple environment (with different resources shared by specific groups of members of the VAXcluster or, possibly, one member providing special-purpose functions).

In any VAXcluster system, users can share computing, disk storage, and batch and print job processing resources. The ability to share resources facilitates work-load balancing because work can be distributed across the cluster. Resources can be added or removed without disrupting normal cluster operation.

Members of a VAXcluster system can share processing resources, data storage, and queues under a single OpenVMS security and management domain. Applications run on one or more CPUs in the cluster, accessing shared resources in a coordinated manner. Although most cluster resources can be shared, user processes and memory are CPU specific, and each member can boot or fail independently.

The software components used to implement VAXcluster communication and resource-sharing functions always run on each member of the cluster. If one member fails, the VAXcluster system continues operating because the components still run on the remaining members. These software components are summarized in Table 3–6.

Table 3–6 VAXcluster Software Components

VAXcluster Component	Function
System Communications Services (SCS) software	Implements intercomputer communication, according to the Digital System Communications Architecture (SCA)
VAXport drivers	Control the communication paths between local and remote ports
Connection manager	Dynamically defines the VAXcluster system and coordinates participation of members in the cluster; uses SCS to provide an acknowledged message delivery service for higher OpenVMS software layers; maintains cluster integrity when computers join or leave the cluster
VAXcluster distributed file system	Allows all computers to share mass storage, whether the storage device is connnected to an HSC subsystem or to a computer; used to provide the same access to disks and files across the cluster that is provided on a standalone OpenVMS computer
Queue manager	Makes batch and print queues available across the cluster

(continued on next page)

Table 3–6 (Cont.) VAXcluster Software Components

VAXcluster Component	Function
Distributed lock manager	Used for synchronization functions by cluster facilities and cluster applications developers; implements system services to provide clusterwide synchronization of access to resources by allowing the locking and unlocking of resource names; provides a queueing mechanism so that processes can be put into a wait state until a particular resource is available; supports clusterwide deadlock detection
MSCP and TMSCP servers	Implement the disk mass storage control protocol and tape mass storage control protocol for communicating with disk and tape controllers, respectively; make locally connected disks and tapes available across the cluster

In addition to these components, all VAXcluster systems require DECnet software, which ensures that system managers can access all VAXcluster members from a single terminal, even if terminal-switching facilities are unavailable.

System managers control how jobs share batch processing and printer resources by setting up and maintaining clusterwide generic queues. A generic queue holds a job that will execute on an execution queue on a specific node when the node is available to process jobs. The strategy for setting up and managing the generic queues determines how well work loads are matched to available resources. The clusterwide queue manager process accesses the clusterwide queue database for all processes in a cluster. Job controllers, user processes, and print symbionts all communicate directly with the centralized queue manager through a shared interprocess communication link.

VAXcluster systems can make disk and tape storage resources accessible to all VAXcluster members. A cluster-accessible storage device can be used directly by multiple members of the cluster. Disks and tapes can be made accessible to members by means of OpenVMS MSCP and TMSCP server software. Cluster-accessible disks offer the following advantages:

- More efficient use of mass storage, because more than one member can use the same disk.
- Access by users to their default work disks when logging in to any member on which the disks are accessible.
- Clusterwide file sharing. Because members can share common versions of files, updates to a file are made only once to a single copy of the file.
- Implementation of clusterwide queues. Batch and print jobs can be processed on any member that has access to the necessary disks.

Some VAXcluster systems include hierarchical storage controller (HSC) subsystems, self-contained, intelligent mass storage subsystems that enable VAXcluster members to share DSA disks and DSA tapes. HSC disk configurations provide flexibility, expansion potential, and maintenance and backup capability.

The OpenVMS distributed lock manager and the distributed file system enable the development of distributed applications that work from the same data, which is itself distributed across the VAX network.

Disk data can be replicated between VAXcluster systems for better read performance and higher availability using volume shadowing, as described in the following section.

3.4.2 Volume Shadowing Software

Volume shadowing software for OpenVMS enhances data availability by duplicating all data written to disk onto compatible disk volumes, called "shadow sets." A shadow set consists of one, two, or three disk volumes of the same model, referred to as shadow set members. The volume shadowing software can read data from any member of the shadow set.

Because volume shadowing simultaneously records data on more than one disk, the duplication of data ensures that data is consistently available. If one disk becomes unreadable because of normal media deterioration, communication path failure, or controller or device failures, processing continues with another disk in the shadow set. The process of shadowing is invisible to end users and applications. Disks can be added or removed without affecting the user or the application.

Volume shadowing is controller independent and supports shadowing of VAXcluster devices. Phase I volume shadowing performs shadowing of disk devices connected to an HSC. Phase II volume shadowing supports many more disk controllers and devices in a wider range of configurations. Using volume shadowing in a VAXcluster system with multiple controllers ensures a high degree of data availability (see Section 7.1.2.1).

Shadow sets are created with the Mount utility or the $MOUNT system service. A disk is added to the shadow set by means of the MOUNT command or the $MOUNT system service. Volume shadowing routines ensure that, within a reasonable time, the newly added shadow set member is made identical to the other member or members of the set. A disk is removed from the shadow set by operator command or automatically, if the disk becomes inoperative.

3.4.3 RMS Journaling Software

RMS Journaling for OpenVMS is an optional software tool that maintains the data integrity of RMS files if any of a number of failures occur. It helps to protect RMS data from becoming lost or inconsistent. RMS Journaling supports distributed transactions through the use of DECdtm (see Section 3.1.6) and provides support for programs that use multiple concurrent transactions. A transaction is a series of RMS record operations made on one or more files. RMS journaling helps prevent data from becoming inconsistent due to the incomplete execution of a transaction.

RMS Journaling provides three forms of journaling:

- After-image journaling provides the means to redo a series of modifications to a data file, enabling the recovery of lost or corrupted files. After-image recovery restores the contents of the file from the point of its latest backup copy.
- Before-image journaling provides the means to undo a series of modifications to a data file, in the event that a file is updated with erroneous data. It also permits automatic rollback to the last consistent state in the event of an error or a failure.
- Recovery-unit journaling maintains transaction integrity by preventing partial completion of transactions.

4 Development on OpenVMS Systems

An essential feature of the OpenVMS operating system is its support of a rich environment for developing software application programs. The programming software integrated in the OpenVMS system provides the tools required to develop new software applications effectively. Also, the developer has the option of using additional powerful tools to enhance the productivity of software development in the OpenVMS environment.

This chapter summarizes the primary features of OpenVMS that support program development. These standard features are available on all OpenVMS systems. The chapter introduces the common programming environment and presents brief functional descriptions of the OpenVMS programming tools, as well as OpenVMS POSIX programming capabilities. The use of optional software development tools running on OpenVMS systems in the context of distributed multivendor environments is covered in Section 6.3.3. For additional information about the OpenVMS programming environment and tools, refer to the *OpenVMS Programming Environment Manual.*

4.1 Common Programming Environment

The OpenVMS system supports a flexible programming environment that offers a wide range of tools and resources to support efficient program development. The common programming environment permits the development of mixed-language application programs and portable programs, as well as application programs with distributed functions that run in client/server environments.

In the common programming environment, programmers can use OpenVMS resources to perform such tasks as:

- Creating, controlling, and deleting processes
- Communicating with other components
- Sharing resources
- Implementing input/output procedures
- Using security features
- Managing memory
- Managing files
- Synchronizing events
- Providing for condition handling
- Calling utility routines

The components of an OpenVMS application are the main program, shared libraries, functional routines, and a user interface. Software tools that support development of applications in the OpenVMS programming environment include:

- Language compilers, interpreters, and assemblers
- Linkers and debuggers
- Text processors and other program development utilities
- Callable system routines such as run-time routines, system services, and other utility routines
- Record Management Services (RMS) routines and utilities

Optional software development tools that run on the OpenVMS system enhance programmer productivity, saving programming time and promoting the development of error-free code. OpenVMS supports optional integrated software products that enhance program development capabilities in an organization. These software development products can make use of the Network Application Support (NAS) services, which facilitate the development of applications for multivendor networks. Optional software development products are discussed in Section 4.7 and use of these optional tools in a distributed multivendor environment is covered in Section 6.3.3.

4.1.1 Programming to Standards

Coding of programs for the OpenVMS environment and for other environments involves conformance to software development standards. OpenVMS standards that define modular programming techniques and procedure calling and condition handling practices pertain to applications specific to OpenVMS. IEEE and international standards apply to applications developed on OpenVMS that are designed to run on other systems as well as OpenVMS. Refer to Appendix A for a list of standards supported by OpenVMS.

4.1.1.1 Common Environment for Writing Code

OpenVMS software programmers can write code in a common environment, following standard OpenVMS modular programming practices. This standard approach establishes the minimum criteria necessary to ensure the correct interface at the procedure level between software written by different programmers. If all programmers coding OpenVMS applications follow this standard approach, modular procedures added to a procedure library will not conflict with other procedures in the library. Standard modular programming practices apply to OpenVMS programs that have a public entry point. For details of this standard approach, see the *Guide to Creating OpenVMS Modular Procedures*.

4.1.1.2 Common Language Environment

The OpenVMS system supports a common language environment, which permits use of a mixture of languages in programming. A program written in any of the programming languages supported by OpenVMS can contain calls to procedures written in other supported languages. Mixed-language programming is possible because all supported languages adhere to the OpenVMS calling standard. This standard describes the techniques used by all VAX languages for invoking routines and passing data between them. It also defines the mechanisms that ensure consistency in error and exception handling routines, regardless of the mix of programming languages in use. Information about the calling standard appears in the *OpenVMS Calling Standard*.

4.1.2 Developing Portable Programs

A software program that is portable can be moved from one computer system to another. OpenVMS programmers can design and develop programs to run on different platforms and execution environments. Primary concerns in designing portable applications include:

- Using modularization and abstraction to organize software
- Avoiding platform-specific features

OpenVMS Portable Operating System Interface (POSIX) software that conforms to IEEE POSIX standards (see Section 2.1.3.1) allows OpenVMS users to develop applications that can be moved to other systems that support POSIX. The source code of an application that conforms to POSIX standards can be ported to another POSIX-conforming system and compiled and linked on that system to produce an executable image. The programming interface for POSIX for OpenVMS VAX is described in Section 4.4 and the user interface in Section 5.3.

4.2 OpenVMS Programming Software

This section describes the integrated programming tools available on the OpenVMS operating system to help implement software development.

The phases of a typical software development life cycle can include proposal of the concept; formulation of requirements and specifications for the software product; design, implementation, and testing of the software; and integration and maintenance of the product. Implementing the software product involves building and modifying source code modules and compiling, linking, and executing the resulting images. Testing involves refining code to optimize performance.

As part of the software development life cycle, OpenVMS operating system components and optional software products running on OpenVMS are used to develop applications. Programming language software supported by OpenVMS is described in Section 4.2.2. Other major OpenVMS programming software is listed in Table 4–1. Optional program development software tools that run on OpenVMS are described in Section 4.7.

Table 4–1 OpenVMS Programming Software

Types of Software	OpenVMS Software Components
Text processors	Digital Standard Editor (EDT) DEC Text Processing Utility/Extensible Versatile Editor (DECTPU/EVE) vi editor (POSIX) DEC Language-Sensitive Editor/Source Code Analyzer (LSE/SCA) Text Editor and Corrector (TECO)
Major programming utilities	Linker OpenVMS Debugger Delta/XDelta Debugger

(continued on next page)

Table 4–1 (Cont.) OpenVMS Programming Software

Types of Software	OpenVMS Software Components
Other program development utilities	Command Definition Utility Librarian utility Message utility Patch utility SUMSLP utility National Character Set utility System Dump Analyzer OpenVMS POSIX utilities (yacc, ar, make, lex)
Callable system routines	Run-time library routines System services Utility routines Record Management Services (RMS) routines and utilities

4.2.1 Creating Program Source Files

OpenVMS text processing utilities can be used to create and modify program source files. The Digital Standard Editor (EDT) is an interactive text editor that provides editing in keypad and line modes. EDT supports multiple buffers, startup command files, and journaling. The DEC Text Processing Utility (DECTPU) is a high-performance text processor that can be used to create text editing interfaces such as EVE. DECTPU includes a high-level procedure language with its own compiler and interpreter, as well as the customizable EVE editing interface. DECTPU features multiple buffers, windows, and subprocesses, and provides for text processing in batch mode. In general, the EVE editing interface offers more capability than EDT for complex editing tasks. (Use of the EDT and EVE editors for editing text files is described in Section 5.5.4.) TECO is a traditional character-oriented text editing program that runs under OpenVMS. It can be used to edit program sources and other ASCII text.

The vi editor is a display-oriented interactive text editor used in the OpenVMS POSIX environment.

Other optional tools for creating source files on OpenVMS systems are available separately or as part of the COHESION software development environment (see Section 6.3.3). DEC LSE provides a multilanguage, multivendor editor for program development and maintenance. DEC SCA supplies cross-referencing features and the capability to analyze source code.

4.2.2 Creating Object Files

OpenVMS supports a variety of optional language compilers, interpreters, and assemblers that translate source code to object code (in the form of object modules). These language implementations adhere to industry standards, including ANSI, X/Open, and POSIX standards as well as U.S. Federal Information Processing Standards (FIPS) and Military Standards (MIL-STD), as applicable. Table 4–2 lists language compilers, interpreters, and assemblers supported in the OpenVMS VAX environment.

Table 4–2 Compilers, Interpreters, and Assemblers

Language	Characteristics
VAX Ada	Complete production-quality implementation of Ada language; fully conforms to ANSI and MIL-STD standards; has Ada validation
VAX APL	Interpreter with built-in editor, debugger, file system, communication facility
VAX BASIC	Used as either an interpreter or a compiler; fully supported by the OpenVMS debugger; fully reentrant code
VAX BLISS-32	Provides advanced set of language features supporting development of modular software according to structured programming concepts; provides access to most VAX hardware features
VAX C	Full implementation of C programming language with added features for performance enhancement in the OpenVMS environment
DEC C++ for OpenVMS VAX	C++ compiler with a new DEC C Run-time Library and debug and LSE support.
VAX COBOL	Compatible with ANSI-standard COBOL; supports an embedded data manipulation language interface to the Digital Database Management System (DBMS) that complies with CODASYL standards
VAX DIBOL	Designed for interactive data processing; includes a compiler, debugger, and utility programs for data handling, data storing, and interprogram communication
DEC FORTRAN for OpenVMS VAX	Supports ANSI-standard FORTRAN–77 with many industry-leading extensions; conforms to FIPS standards; has a high optimization compiler and takes full advantage of features of OpenVMS environment
VAX LISP	Fully interactive interpreter and compiler; includes its own debugger
VAX MACRO	Assembly language for programming the VAX computer under the OpenVMS operating system; uses all OpenVMS resources; supports large instruction set enabling complex programming statements
VAX OPS5	Highly efficient language for implementing expert systems; used to apply artificial intelligence technology to production systems
VAX Pascal	Supports standard ANSI Pascal features and added features using VAX floating-point hardware, character instruction sets, OpenVMS virtual memory
VAX PL/I	Includes a compile-time preprocessor that allows language extension and conditional compilation
VAX RPG II	Supports industry-standard RPG II specifications; full access to OpenVMS resources
VAX SCAN	High-level programming language designed to manipulate text strings and text files; used in applications as preprocessors, filters, translators, extractors/analyzers

(continued on next page)

Table 4–2 (Cont.) Compilers, Interpreters, and Assemblers

Language	Characteristics
DEC Trellis	Object-oriented language; batch or incremental compiler with completely integrated tool set (browsers, development tools, debugger)

In addition, OpenVMS serves as the host operating system for Ada cross-compilers, cross-assemblers, and cross-development tools that conform to MIL-STD standards.

4.2.3 Creating Runnable Programs

After a program source file is coded, it must be compiled or assembled into object modules by a language processor and then linked. The OpenVMS Linker binds the object modules into an image that can be executed on the OpenVMS operating system.

The linker processes object modules and shareable image files, as well as symbol table files, library files, and options files (used to manage the linking operation and simplify the use of complex, repetitious linker operations). The most common output of the linker is an executable image of the program. The linker can also produce a shareable image, a system image, an image map, or a symbol table file to be used by other programs being linked.

The Librarian utility provides for efficient storage in central, easily accessible files of object modules, macros, help text, or other record-oriented information.

4.2.4 Testing and Debugging Programs

The debugger allows users to trace program execution and to display and modify register contents using the same symbols as are in the source code.

Two debugger utilities are available on the OpenVMS operating system:

- The OpenVMS Debugger (debugger), which debugs user-mode code
- The Delta/XDelta Debugger (DELTA/XDELTA), which debugs code in other modes as well as user mode

The OpenVMS symbolic debugger is more useful than DELTA/XDELTA for most programs: the symbolic debugger employs user-defined symbols referring to program locations, accepts commands entered using different interfaces (keypad, command line, or file of commands), displays source code lines on the screen, has more descriptive error messages, and provides help information.

The debugger command language includes more than 100 commands to control a debugging session, including these tasks:

- Control program execution on line-by-line basis or at a user-specified breakpoint
- Display breakpoints, tracepoints, watchpoints, active routine calls, stack contents, variables, symbols, source code, and source directory search list
- Define symbols
- Create key definitions
- Change values in variables

- Evaluate a language or address expression
- Create or excute debugger command procedures

The OpenVMS symbolic debugger also provides enhanced support for programs that have multiple threads of execution within an OpenVMS process, including any program that uses DECthreads or POSIX 1003.4a thread services for developing real-time applications.

4.2.5 Using Other Program Development Utilities

Other OpenVMS utility programs used for program development are listed in Table 4–3. RMS utilities, which permit file analysis and tuning, are covered in Section 4.8.2.

Table 4–3 Other OpenVMS Program Development Utilities

Utility	Function
Command Definition Utility	Enables an application developer to create commands with a syntax similar to DIGITAL Command Language (DCL) commands
Message utility	Permits user to create application messages to supplement the OpenVMS system messages
Patch utility	Permits users to make temporary changes (in the form of patches) to an image file; the new version can then be run without recompiling and relinking
SUMSLP utility	A batch-oriented editor used to make several updates to a single source file; one update program can be applied to all versions of a file
National Character Set utility	Permits users to define non-ASCII string collating sequences and to define conversion functions; allows an RMS indexed file to be collated using user-specified collating sequences
System Dump Analyzer utility	Used to determine the cause of system failures; reads the crash dump file and formats and displays it; also used to diagnose root causes that lead to an error

4.2.6 Managing Software Development Tasks

Optional products that run on OpenVMS systems can be used to manage the complexity of software development tasks:

- VAX DEC/Code Management System (VAX DEC/CMS) provides an efficient method of storing project files (such as documents, object files, and other records) and tracking all changes to these files.
- VAX DEC/Module Management System (VAX DEC/MMS) automates building of software applications.

These products are also available as part of the optional COHESION software development environment (see Section 6.3.3).

The optional POSIX for OpenVMS product supports the make utility, which is used to build an application in the OpenVMS POSIX environment. This utility is analogous to DEC/MMS in the OpenVMS environment.

4.3 Using Callable System Routines

OpenVMS provides extensive libraries of prewritten and debugged routines that can be accessed by programs. Libraries specific to OpenVMS that supply commonly needed routines optimized for the OpenVMS environment include run-time library routines, system services, utility routines, and RMS services. These libraries are described in this section.

OpenVMS also supports programming for an open environment with libraries of industry-standard routines. Examples are libraries of optional routines for software products like DECwindows Motif, CDA (Compound Document Architecture), and PHIGS and GKS graphics products.

Another major library of run-time routines supported on OpenVMS systems are DECthreads services. DECthreads services are based on the IEEE POSIX draft standard 1003.4a and are compliant with OSF Distributed Computing Environment standards (see Section 2.2.4). DECthreads routines provide a set of portable services that support concurrent programming by creating and controlling multiple threads of execution within the address space provided by a single process on an OpenVMS system. DECthreads run-time library services provide an application programming interface usable from all OpenVMS languages and an open C-only interface that conforms to the POSIX pthreads standard.

4.3.1 Using Run-Time Library Routines

The OpenVMS Run-Time Library (RTL) is a set of language-independent procedures for programs to be run specifically in the OpenVMS environment. RTL routines establish a common run-time environment for application programs written in any language supported in the OpenVMS common language environment. RTL procedures adhere to the OpenVMS calling standard and can be called from any program or program module in a language supported by OpenVMS (see Section 4.2.2).

The run-time library provides general-purpose functions for application programs. Examples of groups of RTL routines are summarized in Table 4–4.

Table 4–4 Groups of OpenVMS Run-Time Library Routines

Routine	Description
LIB$ routines	Library routines that perform generally needed system functions such as resource allocation and common I/O procedures
MTH$ routines	Math routines that perform arithmetic, algebraic, and trigonometric functions
STR$ routines	String manipulation routines
SMG$ routines	Screen management routines used in design of complex images for character-cell terminals
PPL$ routines	Parallel processing routines
OTS$ routines	Language-independent routines that perform tasks such as data conversion

In addition, language-specific RTL routines support procedures in Ada, BASIC, C, COBOL, FORTRAN, Pascal, PL/I, and RPG, as well as in POSIX C.

4.3.2 Using OpenVMS System Services

OpenVMS system services are procedures that control resources available to processes, provide for communication among processes, and perform basic operating system functions such as I/O coordination. Application programs can call OpenVMS system services to perform the same operations that the system services provide for the OpenVMS operating system (for example, creating a process or subprocess).

At run time, an application program calls a system service and passes control of the process to it. After execution of the system service, the service returns control to the program and also returns a condition value. The program analyzes the condition value, determines the success or failure of the system service call, and alters program execution flow as required.

OpenVMS system services are divided into functional groups, as shown in Table 4–5. System services can be used to protect and fine-tune the security of the OpenVMS environment, handle event flags and system interrupts, designate condition handlers, and provide logical name services and timer services to the application. Other system services control and provide information about processes, manage virtual memory use, and synchronize access to shared resources.

Table 4–5 Groups of OpenVMS System Services

Service Group	Function
Security	Mechanisms to enhance and control system security
Event flag	Clear, set, and read event flags; place process in wait state until flags are set
AST	Control handling of software interrupts called asynchronous system traps (ASTs)
Condition handling	Designates condition handling procedures that gain control when an exception/condition occurs
Logical	Provide a generalized logical name service
Time and time conversion	Permits scheduling of program events at specific times or time intervals; supplies binary time values
Process control	Create, delete, and control the execution of processes (on a clusterwide basis)
Process information	Provides information about processes
Memory management	Permits control of an application program's virtual address space
Change mode	Changes the access mode of a process
Lock management	Permits cooperating processes to synchronize their access to shared resources
Input/output	Perform input and output operations directly at the device driver level, bypassing RMS
DECdtm services	Provides for complete and consistent execution of distributed transactions and for data integrity

OpenVMS I/O system services perform logical, physical, and virtual I/O and network operations, and queue messages to system processes. The $QIO system service provides a direct interface to the operating system's I/O routines. These services are available from within most programming languages supported by OpenVMS and can be used to perform low-level I/O operations efficiently with a minimal amount of system overhead for time-critical applications. (OpenVMS I/O subsystem software is described in Section 3.1.1.)

DECdtm services ensure consistent execution of applications on the OpenVMS operating system. In transaction processing applications, many users may be simultaneously making inquiries and updating a database. The distributed transaction processing environment typically involves communication between networked systems at different locations. DECdtm services coordinate distributed transactions by using the two-phase commit protocol and implementing special logging and communication techniques. DECdtm services ensure that all parts of a transaction are completed or the transaction is aborted. (The two-phase commit protocol supported by DECdtm is discussed in Section 3.1.6 and Section 7.2.3.)

4.3.3 Using OpenVMS Utility Routines

OpenVMS programs can access some OpenVMS utilities through callable interfaces. Utility routines enable programs to invoke the utility, execute utility-specific functions, and exit the utility, returning to the program. Examples of OpenVMS utility routines appear in Table 4–6.

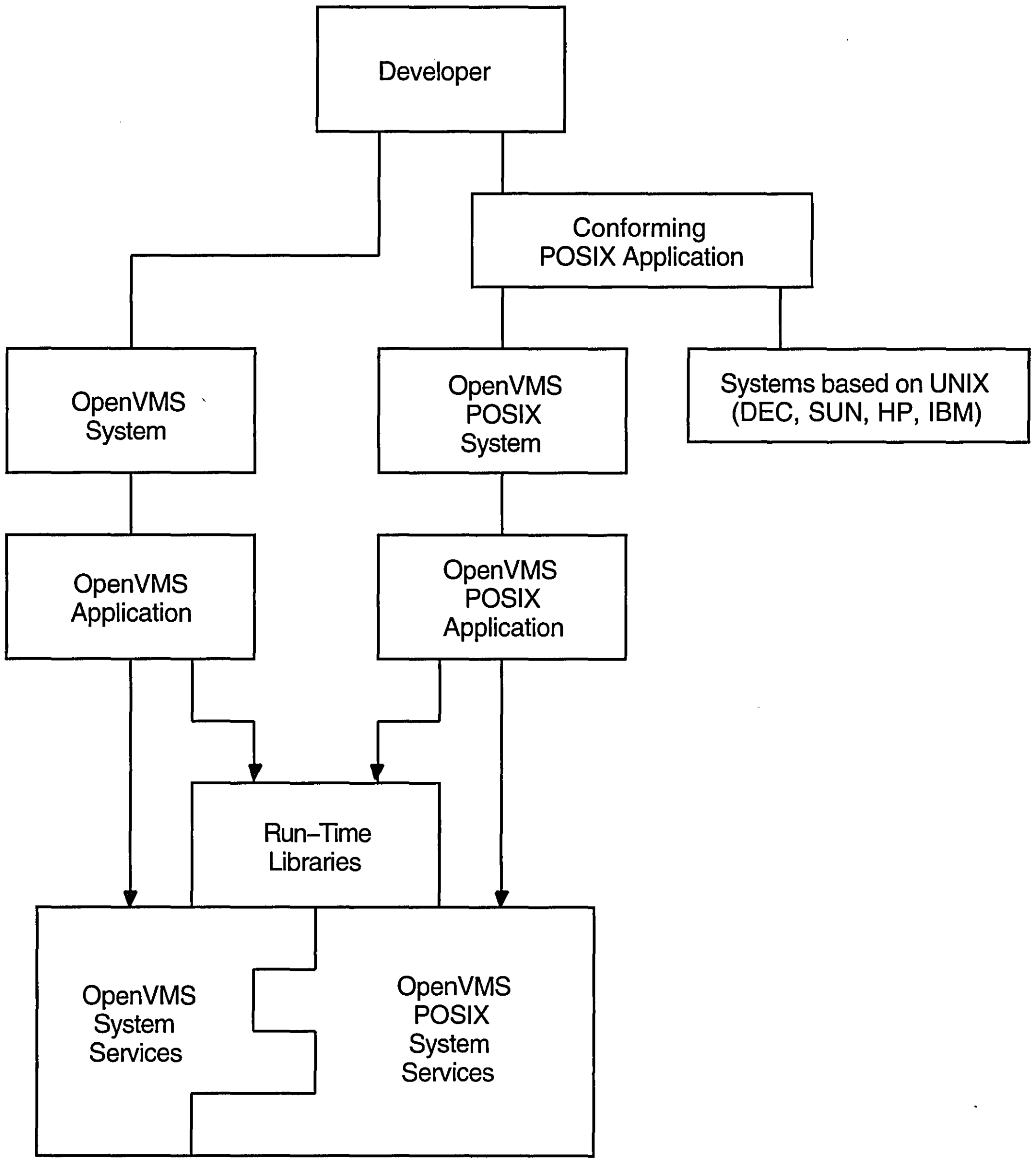
Table 4–6 OpenVMS Utility Routines

Routine	Utility/Facility
ACL$	Access control list (ACL) editor
CLI$	Command Definition Utility (CDU)
CONV$	Convert and Convert/Reclaim (CONV) utility
DCX$	Data Compression/Expansion (DCX) facilty
EDT$	EDT editor
FDL$	File Definition Language utility (FDL)
LBR$	Librarian utility (LBR)
MAIL$	Mail utility
NCS$	National Character Set utility (NCS)
PSM$	Print Symbiont Modification (PSM)
SMB$	Symbiont/Job-Controller Interface (SMB) facility
SOR$	Sort/Merge (SOR) utility
TPU$	DEC Text Processing Utility (DECTPU)

4.4 POSIX Programming on an OpenVMS System

The POSIX for OpenVMS product offers customers the capability to develop and run open, portable applications on OpenVMS VAX. Applications written to POSIX and X/Open standards and draft standards (see Section 2.1.3.1) are portable across a wide range of systems that support those same standards, including systems that do or do not support UNIX. Application developers can develop and deploy their applications on any system that conforms to POSIX standards, including OpenVMS VAX (see Figure 4–1). Applications that strictly conform to the POSIX standards can be developed on the OpenVMS system with OpenVMS POSIX and then ported without modification to any other platform that supports the same POSIX standards. Applications developed on platforms other than OpenVMS that strictly conform to the same POSIX standards and drafts supported by OpenVMS POSIX can be ported and run on an OpenVMS system on which OpenVMS POSIX is installed.

Figure 4–1 Developing POSIX Applications


Developer
Conforming POSIX Application
OpenVMS System
OpenVMS POSIX System
Systems based on UNIX (DEC, SUN, HP, IBM)
OpenVMS Application
OpenVMS POSIX Application
Run–Time Libraries
OpenVMS System Services
OpenVMS POSIX System Services

ZK–5482A–GE

4.4.1 POSIX Applications Using OpenVMS Capabilities

Applications conforming to POSIX standards can coexist and interoperate with traditional OpenVMS applications on the same system. POSIX applications, when run in the OpenVMS environment, have the option of taking advantage of OpenVMS capabilities such as VAXclusters and volume shadowing, as well as file and data protection, security, and SMP. POSIX applications can also optionally call OpenVMS services and libraries, such as file journaling. In addition, POSIX applications can use other standards-based libraries such as Motif, GKS, PHIGS, and SQL.

OpenVMS POSIX users can log in directly to the OpenVMS DCL environment or to the OpenVMS POSIX interactive shell environment (see Section 5.3). OpenVMS POSIX program developers who work at the DCL level can use the OpenVMS program development environment tools, including the VAX C compiler, the OpenVMS Linker, and OpenVMS Debugger and their choice of editor, as well as optional application building tools that run on OpenVMS. Developers who work at the OpenVMS POSIX shell level can use the environment defined in POSIX draft standard P1003.2 for linking and compiling programs, building applications, and archiving library entries. They can also use the vi utility defined in POSIX P1003.2a and other software development utilities defined in the POSIX P1003.2 draft.

4.4.2 OpenVMS POSIX Programming Interface

OpenVMS POSIX application programs are written using the C language and functions defined by the POSIX and X/Open standards and draft standards.

OpenVMS POSIX supports the POSIX 1003.1 standard, which incorporates ANSI C and includes a series of system services. These services allow users to perform operations such as process creation and execution, file system access, and I/O device management. POSIX system services supported by OpenVMS POSIX include:

- Process creation, execution, and termination functions
- Process environment functions
- A series of POSIX functions that provide for file and directory operations
- I/O functions that include file I/O and creation of a pipe that serves as an interprocess channel
- Terminal interface functions involving mapping of a set of control character functions to sets of keys that are UNIX style or OpenVMS style
- Header files used for POSIX applications

In an OpenVMS POSIX environment, a new process is created using the fork function, which generates a child process that can then independently execute a program using the exec family of functions. (The child process is analogous to an OpenVMS subprocess with some basic differences; if the parent process terminates, the POSIX child process can continue to exist, while the OpenVMS subprocess would terminate.) In POSIX, abnormal termination of a process can occur because a signal is received by the process, informing the process that an event has taken place. Each signal causes the application to take an action (the signal is analogous to an asynchronous system trap [AST] in an OpenVMS environment). POSIX includes alarm, pause, and sleep functions that affect the process.

OpenVMS POSIX supports the POSIX P1003.2 draft standard, which includes an interactive interface (described in Section 5.3) and a callable interface to POSIX shell and utility services. The interactive interface to OpenVMS POSIX is a set of commands and utilities similar to UNIX commands and utilities with many of the same functions and features of the Korn shell. These commands and utilities include those that provide functions similar to DCL (see Section 5.5.3) in addition to functions not available in the DCL environment. (The use of DCL commands is mentioned in Section 4.5 and described in Section 5.2.1.)

OpenVMS POSIX supports a number of utilities and features based on UNIX that are unavailable in the basic DCL environment, including:

- Pipes, which allow the output of one command to become the input to the subsequent command. (In DCL, this process often requires the use of a temporary output file.)
- Complex utilities:
  - The sed utility: A batch steam editor useful for editing extremely large files or making a change to a group of files according to a script of editing commands.
  - The awk utility: A batch stream editor that executes specified actions based on test patterns, using its own syntax (similar in many respects to a C-like syntax); used interactively or in batch mode.
  - The bc utility: An arbitrary precision arithmetic calculation facility that uses a C-like syntax; used interactively or in batch mode.
  - The lex utility: A lexical analyzer generator that reads a description of lexical syntax from input files and generates C language code that performs lexical analysis; this C code can be used by the yacc utility.
  - The yacc (yet another compiler compiler) utility: A tool for writing compilers and command interpreters that parse input according to specific grammar rules, creating tables that are used with C code to constitute a parser that will recognize the grammar.
  - The c89 utility: The POSIX interface to the compiler and linker (analogous to the cc command in UNIX).
- Utilities that facilitate the building of existing UNIX applications that conform to POSIX on an OpenVMS POSIX system, and provide the interface between the UNIX program development environment and the tools available in the OpenVMS environment (VAX C compiler, OpenVMS Linker, OpenVMS Debugger):
  - The ar utility: Used to create and maintain libraries.
  - The ln utility: Used to compile and link application programs.
  - The make utility: Used to build an application (analogous to DEC/MMS in the OpenVMS environment).
- The vi text editor, which is a display-oriented editor familiar to UNIX users.

A set of callable interfaces can be used to execute shell commands, compile and execute regular expressions, and perform pattern matching.

OpenVMS POSIX implements the POSIX P1003.4 draft standard, which defines a set of real-time functions. For applications that have real-time computing requirements, these extensions provide support for such functions as enhanced interprocess communication, scheduling and memory management control, and asynchronous I/O operations. The following categories of real-time functions are supported:

- Binary semaphores, synchronization mechanisms to control access to systemwide resources
- Interprocess communication between multiple processes; support in the areas of event notification, message queues, and shared memory

- Asynchronous event notification, a way of passing data within an application
- Clocks and timers that let the application set the systemwide clock

In addition, OpenVMS POSIX supports the BASE specifications described in XPG3 (the X/Open Portability Guide Issue 3), implementing the minimum set of components required to create the XPG3 Common Applications Environment. This set of components consists of the internationalized system calls and libraries, commands and utilities, and the C language as implemented on the VAX hardware platform. XPG3 features supported by POSIX for OpenVMS are listed in Table 4–7.

Table 4–7 XPG3 Features Supported by POSIX for OpenVMS

Feature	Utility or Function
XPG3 utilities	lp, lpsat, and cancel utilities
	ctags utility (for development)
	Enhanced make utility
	egrep utility
	Internationalization utility: iconv
	Localization utility: gencat
System routines[1]	Internationalization functions
	Mathematical functions
	Regular expression parsing
	Table and list management: creation, search, and deletion functions
	Other minor functions

[1]A total of 49 CRTL routines

XPG3 defines the run-time behavior of the library routines and how this behavior is affected by the internationalization environment, but does not specify the tools to create and maintain the environment itself. POSIX draft standard P1003.2 defines the following tools for the internationalization environment:

- localdef: Used to define the internationalization environment
- locale: Used to query any internationalization environment currently available on the system

XPG3 internationalization features supported by OpenVMS POSIX allow users to develop applications that can be deployed (without recompiling) in multiple cultures or countries in such a way that users can experience their own language and cultural context. The application developer can use these features to ensure that messages from the application are generated in the local language context.

For additional information about OpenVMS POSIX programming interfaces, refer to the *Guide to Programming with VMS POSIX* and other documentation in the POSIX for OpenVMS documentation set.

4.5 Programming User Interfaces

User interfaces to the OpenVMS operating system include the DCL interface (see Section 5.2.1) and the optional DECwindows Motif graphical user interface (see Section 5.2). Another user interface is through electronic forms (see Section 5.4.1).

DCL commands can be used to invoke program development software (compilers, editors, linkers) and to run and control execution of programs. DCL command procedures can be used to perform repetitious operations in software development.

The Command Definition Utility enables application developers to create DCL-level commands with a syntax similar to OpenVMS DCL commands. Using CDU, the developer can create applications with user interfaces similar to those of operating system applications. The Message utility permits the application developer to create application messages to supplement the system messages supplied by the OpenVMS operating system.

The DECwindows Motif software provides a consistent user interface for developing software applications and includes an extensive set of programming libraries and tools. The applications programmer can use the following DECwindows Motif software to construct a graphical user interface:

- A user interface toolkit composed of graphical user interface objects (widgets and gadgets); widgets provide advanced programming capabilities that permit users to create graphic applications easily; gadgets, similar to widgets, require less memory to create labels, buttons, and separators
- A user interface language to describe visual aspects of objects (menus, labels, forms) and to specify changes resulting from user interaction
- The OSF/Motif Window Manager, which allows users to customize the interface

The DECwindows Motif programming libraries provided include:

- Standard X Window System libraries such as Xlib and the intrinsics
- Libraries needed to support the current base of XUI applications
- OSF/Motif toolkit support for developing applications using the Motif user interface style
- Digital libraries that give users capabilities beyond the standards

Also available in the DECwindows Motif environment are LinkWorks services for creating, managing, and traversing informational links between different application-specific data. LinkWorks services, with the LinkWorks Manager application, help organize information into a hyperinformation environment (see Section 5.2.2). Linkworks Developer's Tools provide a development environment for creating, modifying, and maintaining hyperapplications.

Another optional development tool for the DECwindows Motif environment is DEC VUIT (Visual User Interface Tool). DEC VUIT is an interactive editor in the WYSIWYG (What You See Is What You Get) style. The editor provides a direct manipulation interface to the OSF/Motif User Interface Language (UIL) compiler for the user interface developer.

4.6 Developing Real-Time Applications

The VAXELN Toolkit is a set of tools that can be used on VAX systems running OpenVMS to develop efficient real-time applications (for example, real-time applications for process control, simulation, or high-speed data acquisition). VAXELN real-time applications are run on rtVAX computers (see Section 1.4.2). The VAXELN real-time operating environment optimizes dedicated real-time systems for predictability and fast response time. VAXELN transparently supports open standards such as IEEE 1003.1 and draft 10 of IEEE 1003.4. VAXELN operates in TCP/IP and DECnet local and wide area networks and participates in the NAS environment.

4.7 Digital Software Development Tools

Digital supplies optional software development tools for the OpenVMS environment, such as DECset. DECset is a set of tools that support software coding, testing, and maintenance of applications and data. These tools can be used individually or as part of the optional Digital COHESION environment for computer-aided software engineering (CASE).

COHESION integrates software to increase an organizations's software development productivity while permitting the choice of platforms, tools, and other elements to remain in the user's hands. The COHESION environment is built on Network Application Support (NAS) software. Standards-based NAS services assist programmers in developing integrated applications with functions that can be distributed across heterogeneous platforms (see Section 2.1.5.2).

Use of COHESION software tools on the OpenVMS system in multivendor distributed environments is described in Section 6.3.3.

4.8 Managing Data

The basic OpenVMS tool for transparent, intuitive management of data is the Record Management Services (RMS) subsystem. RMS is a collection of routines that give programmers a device-independent method for storing, retrieving, and modifying data for their application. RMS also provides extensive protection and reliability features to ensure data integrity.

RMS is a higher level interface to the file system and OpenVMS I/O subsystem. It is used by all products that run on OpenVMS for file and record operations. A subset of RMS services permit network file operations that are generally transparent to the user.

4.8.1 RMS Files and Records

RMS supports a variety of file organizations, record formats, and record-access modes. RMS supports sequential, relative, and indexed disk file organizations, and fixed- and variable-length records. It supports a number of record-access modes: sequential, by key value, by relative record number, or by record file address. RMS is designed primarily for mass storage devices (disks and tapes), but also supports unit-record devices such as terminals or printers.

RMS routines assist user programs in processing and managing files and their contents. RMS routines perform these services for application programs:

- Creating new files, accessing existing files, extending disk space for files, closing files, and obtaining file characteristics
- Getting, locating, inserting, updating, and deleting records in files

RMS promotes safe and efficient file sharing by providing multiple access modes, automatic record locking when applicable, and optional buffer sharing by multiple processes.

RMS files are also used in the OpenVMS POSIX environment. Files created by POSIX applications on OpenVMS are binary stream RMS files. POSIX developers can handle files in alternative ways. In the OpenVMS environment, the OpenVMS POSIX developer can create files using OpenVMS naming techniques and then use DCL to access all files generated by OpenVMS or POSIX. In the OpenVMS POSIX environment, to obtain full POSIX file compliance, a container file system is required as part of the POSIX filename space. A container file system is an RMS directory with an extra file to map between POSIX filenames and RMS filenames. The container file system is useful in implementing a file intended to be used on multiple networked platforms, including NFS file systems.

For a description of OpenVMS POSIX files and directories, see Section 5.5.2.

4.8.2 RMS Utilities

RMS file utilities allow users to analyze the internal structure of an RMS file and to determine the most appropriate set of parameters to tune an RMS file. RMS utilities can also be used to create, efficiently load, and reclaim space in an RMS file.

RMS file maintenance utilities include the following:

- Analyze/RMS_File utility
- File Definition Language utilities (Create/FDL and Edit/FDL)
- Convert and Convert/Reclaim utilities

The Analyze/RMS_File utility permits the programmer to analyze the internal structure of an OpenVMS RMS file and generate a report on its structure and use, as well as interactively explore the file's structure. The utility can generate an FDL file from an RMS file for use with the Edit/FDL utility to optimize the data file.

File Definition Language (FDL) is a special-purpose language for specifying file characteristics; it is useful with higher level languages or for ensuring that files are properly tuned. FDL makes use of RMS control blocks: the file access block (FAB), the record access block (RAB), and the extended attribute block (XAB).

The Edit/FDL utility creates a new FDL file according to user specifications. The Create/FDL utility uses the specifications of an existing FDL file to create a new empty data file.

The Convert utility can be used to copy records from one file to another, while changing the record format and file organization, and to append records to an existing file. The Convert/Reclaim utility reclaims empty bucket space in an indexed file to allow new records to be written to it.

4.8.3 Database Products

DEC DBMS and DEC Rdb (which includes the components SQL and SQL/Services) are optional database products that run on the OpenVMS system. For a description of database capabilities on OpenVMS systems in multivendor distributed environments, refer to Section 7.3.

5 User Interfaces to the OpenVMS System

The OpenVMS system offers users consistent, easy-to-use interfaces: a natural language command interface and an optional standards-based graphical user interface, as well as specialized forms-based interfaces. People throughout an organization or activity can use these consistent interfaces to access authorized information and resources anywhere in the OpenVMS software environment.

Users can access the OpenVMS operating system from a wide variety of devices, ranging from desktop devices to terminals connected to large computer complexes. Depending on the configuration, OpenVMS users can share in the full capabilities of OpenVMS systems and VAXclusters and access resources on other computers throughout a worldwide multivendor network.

OpenVMS support for open standards permits development of user-portable applications. A user-portable application provides the user with a consistent interface when the application is run on any system (including systems from other vendors) that conforms to the same open standards. The user experiences the same environment from system to system, without the requirement for retraining. OpenVMS support for open, portable software capabilities is discussed in Section 2.1.4.

This chapter describes user access to OpenVMS systems, the different user interfaces, and the various software environments users can access. The chapter also covers OpenVMS support for such general user activities as file handling and electronic mail.

5.1 OpenVMS Operating System Access

Access to the OpenVMS system is controlled by means of user accounts. To log in and gain access to the OpenVMS system, the user supplies the user name and password specified in the appropriate user account. Information about each user account is maintained in the user authorization file (UAF), which can be modified by the OpenVMS system manager (see Section 3.2.2).

The system manager can determine the user's needs and set up an account that controls the user's access to system resources. In establishing an account for an individual user, the system manager assigns a unique user name, password, and user identification code for the account, and specifies the system privileges and resource quotas associated with the account. Examples of protection mechanisms the system manager can use to control access to system objects (such as files) include the following:

- User identification code (UIC) that identifies the user as a member of a group that can share specific data
- Access control list (ACL) that permits specific users to be included or excluded from accessing a file or achieving certain kinds of access (for example, read or write access)

For additional information about user accounts and other protection mechanisms employed to control user access to the system, see Section 3.3.2.

Some users can access a turnkey or "captive" account without supplying a user name or password. For example, turnkey accounts can be established for special environments such as a factory floor, in which users perform routine tasks and do not need to submit commands to the operating system. Special limited-access accounts are described in Section 3.3.2.

If a user is logged in to an OpenVMS system connected to a network and has access to an account on another node (which need not be an OpenVMS node) in the network, the user can log in to that account from the local node and use the facilities of that remote node while remaining physically connected to the local node. For example, an OpenVMS user can issue the SET HOST command to access a remote node and then log in to the remote node.

Optionally, the user can enter different user environments when logging in to the OpenVMS system. The particular environment is established by the user account and by the way in which the user accesses the system. Possible user environments include:

- The OpenVMS environment (if neither the user nor the system manager specifies an alternative login environment)
- The OpenVMS POSIX environment (if either the user or the system manager explicitly specifies it)
- A customized forms-based environment, such as the ALL–IN–1 integrated office environment (if either the user or the system manager specifies it)

The OpenVMS POSIX environment supported by the OpenVMS system is similar to a UNIX environment (see Section 2.1.3.1). POSIX is designed to enable users and applications that comply with POSIX standards to move between systems. The OpenVMS POSIX user environment is described in Section 5.3.

Optional user interfaces based on forms are described in Section 5.4.

5.2 OpenVMS User Environment

When a user logs in to the OpenVMS operating system, the system creates an environment from which the user can enter commands and run programs. The environment is called a process. The characteristics of the process are specified in the user authorization file (UAF) set up for the user. Within the process, the system executes programs (called images or executable images) one at a time. An image is a file that contains machine-readable instructions and data. The system and the user can both supply image files.

Users can access OpenVMS through the DIGITAL Command Language (DCL) interface supplied with the system (described in Section 5.2.1) or the optional DECwindows Motif graphical windowing interface (described in Section 5.2.2).

When a user logs in to the OpenVMS system, two DCL command procedures are automatically executed: a system login command procedure that the system manager sets up and a personal login command procedure that either the system manager or the user can set up. Editing the personal login command procedure permits the user to customize the OpenVMS computing environment. (DCL command procedures are described in Section 5.2.1.1).

5.2.1 DCL Command Language Functions

DCL provides the user with a consistent user interface. The command language is a set of English-like instructions that tell the OpenVMS system to perform specific operations. The DCL command takes the form of a command name followed by parameters and qualifiers, as appropriate. OpenVMS users can use DCL to conduct a dialogue with the system or to initiate system utilities or user programs.

Users on character-cell terminals can enter DCL commands interactively. OpenVMS users on workstations can enter DCL commands using the DECterm window in the DECwindows Motif interface. OpenVMS users can also include DCL commands in command procedures, invoking the command procedures interactively or submitting them in batch queues for deferred execution. DECwindows users can use the "point and click" technique to execute DCL commands, using the FileView window.

The range of tasks that can be performed using DCL commands is extensive. For example, DCL commands permit the user to:

- Get information about the system
- Modify the work environment
- Work with files
- Work with devices (such as disks and magnetic tape)
- Develop and execute programs
- Provide security and ensure that resources are used efficiently

Table 5–1 lists types of computing tasks and examples of the DCL commands used to perform the tasks. DCL commands are described in alphabetic order in the *OpenVMS DCL Dictionary*.

Table 5–1 Types of Tasks Performed by Commonly Used DCL Commands

Types of Tasks	Examples of DCL Commands
General session control and environmental control	HELP, SHOW, SET, ASSIGN, DEFINE, DEASSIGN, PHONE, MAIL, LOGOUT, REQUEST
Volume and device resource control	MOUNT, INITIALIZE, DISMOUNT, ALLOCATE, DEALLOCATE
Program development and execution control	LIBRARY, LINK, RUN, DEBUG, SPAWN, STOP, SUBMIT, SYNCHRONIZE, WAIT, CANCEL
File manipulation control	DIRECTORY, CREATE, EDIT, DELETE, PURGE, RENAME, COPY, APPEND, DIFFERENCES, SORT, OPEN, CLOSE, READ, WRITE, PRINT, TYPE, DUMP, ANALYZE

Users can also use special DCL commands for other purposes, such as customizing the OpenVMS user environment to make it more productive. Certain DCL commands define short, meaningful logical names for commonly used files, directories, and devices. The system also creates a set of systemwide logical names when the user logs in. Other commands create symbols that represent DCL commands, logical values, or numeric or character values that can be combined into expressions to manipulate the values the symbols represent. Lexical functions permit the user to obtain information from the system about system processes, queues, and user functions. Lexical functions are essentially calls to OpenVMS system services.

For detailed information about using DCL, see the *OpenVMS User's Manual.*

5.2.1.1 DCL Command Procedures

Command-level programming permits the user to create DCL command procedures using a text editor or the DCL command CREATE. A command procedure comprises DCL commands to be executed, data lines that are used by those commands, and optional comment lines. Command procedures can be started interactively or by another command procedure. Command procedures can also be run as batch jobs in a batch queue. Using batch mode permits submission of work for execution at a specified time (for example, during off hours). An advantage of batch mode is the ability to restart the procedure if there is a system failure: DCL provides a /RESTART option that causes the system to restart a job if the system crashes before it is completed.

DCL command procedures provide a full programming language for operator and user interfaces. Command procedures and lexical functions can be used to automate routine system management tasks.

Within command procedures executed at the local node, the user can use DCL commands to open and close local and remote files, and read and write records to those files. The user can also submit DCL command procedures residing on remote nodes for execution as batch jobs on those nodes.

5.2.1.2 OpenVMS Help System

DCL and OpenVMS utilities have a consistent integrated help subsystem to provide operational information on all aspects of system operations. The DCL HELP command permits the user to obtain information about using the system, available DCL commands, and system utilities. The online help system includes summary operational information on all aspects of OpenVMS system operation.

In addition, the HELP/MESSAGE utility allows users to access instant online descriptions of system messages at a character-cell terminal or a DECterm window in a DECwindows Motif interface. Operating through the DCL interface, this utility can provide information about messages supplied by OpenVMS, by optional products running on OpenVMS, and by the customer, if desired. For information about HELP/MESSAGE, refer to *OpenVMS System Messages: Companion Guide for Help Message Users*.

5.2.2 OpenVMS DECwindows Motif Interface

OpenVMS users on workstations can access the optional product DECwindows Motif, an open graphical windowing interface enhanced by a set of integrated desktop applications. The DECwindows Motif graphical user interface complies with the Motif specification developed by the Open Software Foundation (OSF) (see Section 2.1.3.3). OSF/Motif specifies both a graphical user interface and an application programming interface. DECwindows Motif is based on the Massachusetts Institute of Technology specification for the X Window System, the de facto standard for distributed windowing systems.

DECwindows Motif supports an interactive, network-based client/server environment. DECwindows applications can run on a client node and display output on an OpenVMS server node. In other words, an application running on a remote machine can display on the user's OpenVMS desktop device, and the user can interact with the application as though it were running locally. Clients and servers communicate with each other by sharing local memory or over DECnet or TCP/IP network connections.

The DECwindows Motif end-user environment involves point-and-click techniques for interacting with the OpenVMS operating system. The user can divide the workstation screen into multiple windows, each representing a different application. DECwindows applications automate many basic tasks (for example, Mail automates sending and receiving interoffice mail).

A workstation mouse or other pointing device permits the user to point to and select text and objects on the screen. Users interact with DECwindows by finding an object on the screen that represents a task, selecting that object, and completing that task. Typical interface objects include:

- Menus, containing items that let the user tell DECwindows what the user wants to do or wants to work with (for example, the Help menu from an application)
- Icons, representing currently running applications that need to remain available but are too large for the screen
- Buttons that affect the positioning of windows on the screen (for example, the minimize button that lets the user shrink a window to an icon and store it as an icon on the screen)
- Dialog boxes that permit the user to supply additional information to the system

Figure 5–1 shows the main DECwindows Motif components. These components include:

- Session Manager: Appears at the start of every DECwindows session and permits the user to control the DECwindows session, customize the DECwindows environment, and run applications.
- FileView: Gives the user a graphical access to DECwindows applications and provides commands for working with files and directories.
- The Workspace: The screen on the workstation and the background of the DECwindows environment. All windows and objects appear on the Workspace.

Table 5–2 lists and briefly describes DECwindows Motif applications.

Figure 5–1 DECwindows Motif User Interface

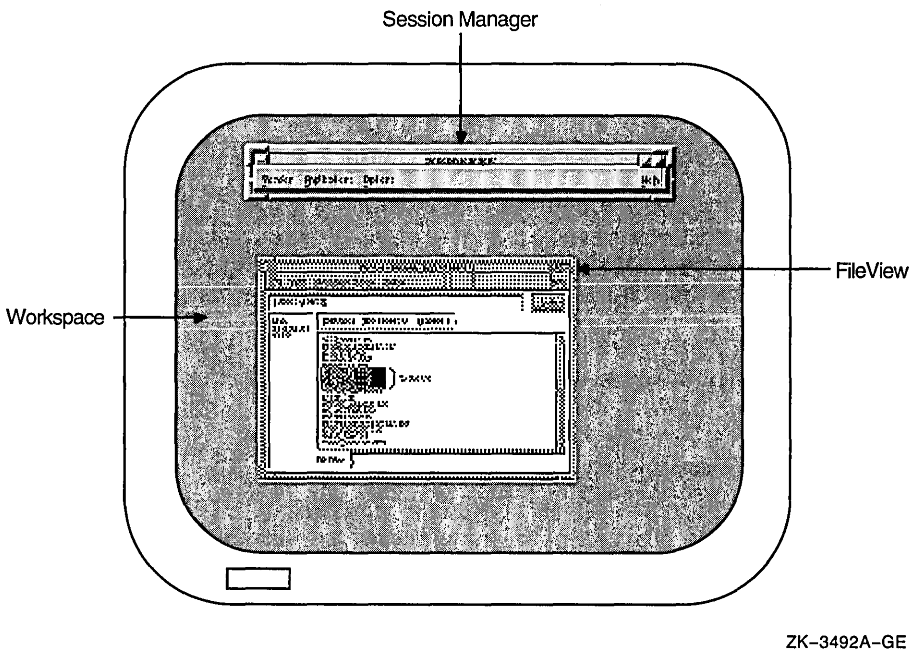


Table 5–2 Commonly Used DECwindows Motif Applications

Application	Function
Bookreader	Displays an online documentation reader on the workstation screen
Calculator	Performs mathematical operations
Calendar	Keeps track of scheduled appointments and assists in planning the user's time
Cardfiler	Organizes information with index cards and card files
CDA Viewer	Displays the contents of many different types of files on the workstation screen (for example, PostScript files)
Clock	Displays the time of day (in both analog and digital format) and the date on the workstation screen with alarm features
DECsound	Records, saves, stores, and plays back audio images
DECterm	Creates a window that emulates a VT300-series terminal
LinkWorks Manager	Controls the hyperinformation environment
Mail	Lets the user exchange messages with other computer users
Paint	Creates a simple picture, an illustration, or a map
Print Screen	Takes a snapshot of the entire screen or a portion of it and prints the file containing the snapshot or saves it
Puzzle	Displays a video version of a number puzzle that contains squares to arrange in ascending order

DECsound is an easy-to-use application that lets the user record and play back audio images on a subset of VAXstation models. DECsound includes sample sounds and provides support for mailing audio messages across the network. It also enables audio messages to be included in compound documents that can be played by the CDA Viewer.

The DECterm terminal emulator provides a traditional character-cell interface, which permits users to enter DCL commmands or use any other command line interface.

LinkWorks is a set of services for creating, managing, and traversing informational links between different application-specific data, such as mail messages, cards from cardfiles, and information from online books. This linked environment is called the hyperinformation environment. LinkWorks services, with the LinkWorks Manager application, help organize information by allowing users to focus on the information rather than on where the information comes from and where it is stored.

DECwindows Motif provides a common "look and feel" across various software environments and tools. The application programmer can construct the user interface using OSF/Motif software tools. The DEC VUIT (Visual User Interface Tool) allows the user to create the OSF/Motif user interface in a visual way, without handcoding the source file. DEC VUIT is supplied with the DECwindows tools package for Motif (see Section 4.5).

5.3 OpenVMS POSIX Environment

A user logging in to an OpenVMS system that includes POSIX for OpenVMS can invoke the POSIX environment, a multiuser timesharing environment supported on character-cell terminals. The environment complies with POSIX draft standard P1003.2a (the *User Portability Extension*), which supports terminal users in a consistent manner across all conforming systems. Users in the POSIX environment need to be familiar with the style of interaction with UNIX systems. Typical users would be program developers, engineers, or general-purpose timesharing users interested in portable applications that are similar to UNIX applications.

The POSIX environment differs from the OpenVMS environment. Commands are limited to those that the POSIX environment understands; commands must be in lowercase characters. The POSIX command prompt is psx>. Some POSIX commands have the same name as DCL commands, but the functions are different. The command line interface includes UNIX features such as pipes, scripts, and complex utilities (see Section 4.4.2).

For detailed information about the OpenVMS POSIX user environment, refer to the *Guide to Programming with VMS POSIX.*

When logging in to an OpenVMS POSIX system, the user can log in directly to the DCL environment or to the OpenVMS POSIX shell environment. The OpenVMS POSIX shell is a command language interpreter (CLI) equivalent to the OpenVMS DCL interpreter. The user can instruct the shell, either interactively or within an OpenVMS POSIX command file (called a shell script), to perform tasks such as calling utilities.

Logging in directly to the POSIX shell environment requires the addition of the qualifier /CLI=POSIX$CLI to the user name. Alternatively, the user can log in to the OpenVMS DCL environment and invoke the OpenVMS POSIX shell.

On an OpenVMS system that includes POSIX for OpenVMS, the user can move back and forth between the OpenVMS and OpenVMS POSIX environments.

5.4 Forms-Based User Environments

A user logged in to an OpenVMS system may interact with application software by responding to electronic forms. An electronic form is a collection of fields and background text displayed on any type of display device. Forms are used with a variety of applications to enhance the gathering and display of information.

The following sections describe the Digital DECforms software product and the forms-based integrated office environment, ALL–IN–1.

5.4.1 DECforms Interface

Some applications that OpenVMS users can access provide user-friendly forms interfaces. The OpenVMS forms software product, DECforms, allows a single application to support multiple types of users with interfaces tailored to their needs, including interfaces in multilingual environments. DECforms supports the full range of OpenVMS terminals and compatible terminal emulators on workstations and personal computers.

DECforms is the OpenVMS implementation of the proposed ANSI ISO standard for a Forms Interface Management System (FIMS), which standardizes the interface between an application and the forms it uses. DECforms offers a subset of the full FIMS functionality, with extensions tailored for the OpenVMS environment. DECforms embodies the fundamental principles underlying the FIMS model:

- Separation of form and function
- Efficient distribution of forms processing
- Ease of use
- Flexibility of user interface control
- Programming language independence

DECforms serves as the user interface into OpenVMS database layered products (DEC Rdb for OpenVMS and DEC DBMS). It combines the capabilities of previous Digital forms systems (VAX FMS and VAX TDMS) and adds new features. DECforms is integrated with the VAX ACMS product to provide powerful forms processing capabilities in transaction processing environments (see Section 7.2.2). A single DECforms run-time process can control multiple terminals simultaneously. Using optional software, DECforms can be distributed to remote CPUs, bringing forms processing as close to the end user as possible. DECforms services are also included in NAS integrated products.

5.4.2 ALL–IN–1 Office Systems Environment

OpenVMS supports an optional, specialized user environment: the forms-based, menu-driven, integrated ALL–IN–1 system. ALL–IN–1 links office applications together and includes a facility for integrating other business-oriented applications. ALL–IN–1, which is easy to use, controls user activities.

When a user who has an ALL–IN–1 user account logs in to an OpenVMS system, the user can then log in to the ALL–IN–1 system. OpenVMS users on workstations can access the ALL–IN–1 system through a dynamic graphical user interface provided by the ALL–IN–1 Services for DECwindows.

ALL–IN–1 communicates with users through forms displayed on the terminal screen. The first form displayed is the Main menu, followed by additional options, each representing a subsystem or specific group of office tasks.

ALL–IN–1 allows users to transport or receive information from other systems through the electronic messaging facility linked to the network. User documents are stored in folders in the user's file cabinet where they are accessed by the office applications. Users can create and process documents using either the EDT or WPS editor or the WPS–PLUS full-function word processor.

ALL–IN–1 is a customizable software product. Using a client/server model to provide the core services that all office workers need, ALL–IN–1 adds advanced capabilities for workgroup computing, enterprise communications, and information management.

5.5 Information Handling on the OpenVMS System

End users on the OpenVMS operating system normally work with information stored in files. OpenVMS supports files that can be used in either the OpenVMS environment or the POSIX environment. In addition, OpenVMS POSIX files can be ported to other systems that conform to POSIX standards.

In the OpenVMS environment, users can enter DCL commands to access and manipulate files and to sort and merge records in files. Software tools are available to create, edit, and process text files. The DECwindows Motif interface also supports file handling through the FileView program, which is a graphical interface to OpenVMS file management (see Section 5.2.2).

5.5.1 OpenVMS Files and Directories

An OpenVMS file can consist of text the user enters and manipulates or machine-readable data that the machine understands. Creating a memo is creating a file; sending an electronic mail message is sending a file. Running a program involves loading an image file into the system and executing the instructions contained in that file.

OpenVMS files are listed in directories. Each directory is a special file that contains the names and locations of files. Every OpenVMS user account has an associated disk directory containing user files. This default or login directory can include many levels of subdirectories, arranged in a hierarchical directory structure.

The full file specification for an OpenVMS file describes the access path the OpenVMS system uses to locate and identify a file. The specification identifies the location at which the file resides: the network node name, device name of the directory disk volume, directory name, and file name, type, and version number of the file. The file type identifies the structure or type of data in the file. (If the node name and directory name are not specified, the defaults are the user's current node and directory name.) The file name can include alphanumeric characters, hyphens, underscores, and dollar signs. The following is an example of a full OpenVMS file specification:

```
BOSTON::USER4:[DATA]SALES.DAT;5
```

Wildcard characters can be used to manipulate large numbers of files without naming them individually (for example, the asterisk can be used to indicate all occurrences of certain fields, or parts of fields, in a file specification).

5.5.2 OpenVMS POSIX Files and Directories

Users in the OpenVMS POSIX environment can choose between using OpenVMS files or POSIX files (similar to UNIX files) for their applications. OpenVMS POSIX users can consider the file system as containing two parts:

- The OpenVMS file system in the context of the OpenVMS POSIX environment. OpenVMS files can be used in the POSIX environment for interoperability with other components of the OpenVMS operating system, but must be referred to by POSIX pathnames.
- The container file system (see Section 4.8.1). The POSIX for OpenVMS container file system permits a file name on an OpenVMS system to be translated to a file that fully supports the POSIX standards and is portable to other systems that conform to POSIX standards.

POSIX files follow rules similar to the UNIX environment. POSIX files are referred to by pathnames that indicate the directory path to the file, including the file name itself (no device name is specified). Subdirectories are preceded by the slash character. The characters in the POSIX file name must be lowercase and can include any character except the slash and the null character. An example of a POSIX pathname is

```
/vms/user/data/sales.dat
```

The OpenVMS file system and the POSIX container file system differ in file specifications, file structures, file protection, use of special files, links between files, and symbolic links.

Software development involving POSIX files is discussed in Section 4.8.1. For additional information about using files in the OpenVMS POSIX environment, refer to the *Guide to Programming with VMS POSIX.*

5.5.3 OpenVMS File Manipulation

In the OpenVMS environment, users can perform file operations by specifying DCL commands. Some of the commands invoke OpenVMS utilities which perform the file operation. File operations performed using DCL commands are summarized in Table 5–3.

Table 5–3 File Operations Performed Using OpenVMS Utilities and DCL Commands

Utility or DCL Command	File Operation
BACKUP utility	Saves and restores user data
CREATE command	Creates files or directories
CONVERT utility	Maintains optimal ISAM file performance, copies records to files of different organization, reformats indexed files
COPY command	Moves files around
DELETE/ERASE command	Erases a file and removes it from a directory
DIRECTORY command	Displays the contents of a current directory
DUMP command	Displays actual file contents to facilitate application debugging

(continued on next page)

Table 5–3 (Cont.) File Operations Performed Using OpenVMS Utilities and DCL Commands

Utility or DCL Command	File Operation
EDIT command	Permits creation of a new file or viewing and changing of the contents of a text file
PRINT command	Sends a specified file to a printer for printing
PURGE command	Facilitates disk space reclamation by deleting old versions of a file
RECOVER command	Applies RMS journals to recover RMS files
RENAME command	Changes the name of a specified file
SET FILE command	Manipulates file characteristics
SORT/MERGE utility	Sorts records in a file and combines previously sorted files into an output file
TYPE command	Displays the contents of a specified file

Users do not normally manipulate the records that comprise a data file, except to sort or merge records in a file. The OpenVMS Sort/Merge utility is used to manage the records. The SORT utility sorts records from 1 to 10 input files into an output file, based on a sequence of user-selected keys in the input files. The user can define key fields on which to organize the files in alphabetic or numeric order, in ascending (low to high) or descending order. The MERGE utility combines up to 10 previously sorted files according to a user-selected key and generates an ordered output file.

OpenVMS users can perform network file operations as a natural extension of the input/output operations performed on their systems. Most DCL commands permit the user to access files on remote systems in the same way as on the local system. Only a node name need be added to the file specification for network operations. Certain commands that invoke processing on a specific system require the /REMOTE qualifier (for example, PRINT/REMOTE and SUBMIT/REMOTE).

The DCL command EXCHANGE/NETWORK allows the transfer of files to or from heterogeneous operating systems that do not support OpenVMS file organizations, over DECnet communications links. For example, users can transfer files between MS–DOS or ULTRIX systems and OpenVMS systems, with the option of modifying file and record attributes.

In the OpenVMS POSIX environment, POSIX interactive users use POSIX P1003.2 commands and utilities to manipulate files. Some POSIX commands perform the same function as DCL commands, but have different names. (For example, the POSIX command ls is equivalent to the DCL command DIRECTORY, and cd is equivalent to the DCL command SET DEFAULT.) Conversely, some POSIX commands have the same name as DCL commands, but the functions are different. (For example, the DCL command TYPE displays a file on the terminal. In an OpenVMS POSIX environment, the command type displays the structure or type and pathname of a command, and either the more or cat utility displays a file on the terminal.)

5.5.4 Text File Editing and Processing

OpenVMS provides several editors for use in editing text files. Text editing is the process of creating and maintaining character-oriented files. Editors can also be used to create and modify source files for programming languages or text formatters (such as VAX DOCUMENT). Two of the most commonly used editors are EDT and EVE:

- Digital Standard Editor (EDT), which permits the user to enter and manipulate text and programs and perform line and character editing. EDT is an easy-to-learn editor, with a help system and a journaling capability to protect against loss of edits. EDT provides screen editing using the keypad on VT-series terminals.
- Extensible Versatile Editor (EVE), which allows you to insert, change, and delete text quickly. EVE is written in the DEC Text Processing Utility (DECTPU) language, described in Section 4.2.1. EVE is a full-screen editor that allows users to scroll through text on a terminal screen. EVE provides an EDT-style keypad.

Another text editor that runs on OpenVMS is the Text Editor and Corrector (TECO), which is used to edit ASCII files.

OpenVMS POSIX supports the vi editor. This editor is a display-oriented interactive text editor that includes the line editor ex. The vi editor is defined in POSIX P1003.2a and in the XPG3 BASE specification.

Text formatters are used to prepare documents, processing source files into formatted text and creating tables of contents and indexes. Two text processors available to OpenVMS users are as follows:

- The DIGITAL Standard Runoff (DSR) facility is a basic text formatter supplied with the OpenVMS system. The user can use a text editor to create a source file with the file type of .RNO and then enter the RUNOFF command to generate a formatted, printable text file with the file type of .MEM.
- VAX DOCUMENT is an optional text formatter that provides the tools for automated book publishing. VAX DOCUMENT software also includes a graphics editor and the capability to produce online documentation for the Bookreader application. Optional products provide for text processing and multimedia document preparation and control on OpenVMS systems; these services are also available with NAS products (see Section 6.3.1).

5.5.5 Electronic Mail

OpenVMS users can communicate with users on their own system and on other systems electronically. The OpenVMS system's communication capabilities are designed to operate over a network as easily as on a single system.

The standard electronic mail facility for OpenVMS users is the VAXmail facility (MAIL), which permits a user to send messages to, and receive messages from, any other user on the same system or to users on other systems on the same network. Once a user is logged in to the OpenVMS system, MAIL displays a message on the terminal screen with a notification of any incoming mail. A user can invoke the Mail utility and, at the MAIL> prompt, send or read mail, or perform other operations such as forwarding or replying to mail, deleting messages, or extracting messages to create text files. A user can also organize a mail file by filing mail in folders and displaying the messages contained in each folder. For more information about using MAIL, refer to the *OpenVMS User's Manual*.

OpenVMS users in the ALL–IN–1 environment can send mail electronically across a heterogeneous network using messaging services based on NAS. DEC MAILworks (formerly known as ALL–IN–1 MAIL) provides access to the full range of Digital multivendor connectivity services by means of the MAILbus facility. The MAILbus family of products links multivendor electronic mail systems and messaging applications into an enterprisewide electronic messaging system.

Other optional communications software permits the OpenVMS user to access the VAX Notes electronic conferencing facility and the DEC Video Text (VTX) facility. VAX Notes supports group conferencing, organizing online discussions into topic-and-reply formats that both encourage and keep a record of team communication. VAX Notes can be accessed from a broad variety of desktop devices. VTX enables users on various desktop devices to access online information (such as documents intended for a wide audience), across the network, using client/server computing. Users find the information they need by responding to a series of menus.

Part III

Open Distributed Computing Environments

This part of the manual describes computing environments in which OpenVMS software can be integrated with software from multiple vendors.

Chapter 6 describes integrating OpenVMS systems with other systems in distributed heterogeneous networks and using NAS software to support multivendor integration. The chapter explains how OpenVMS systems and VAXcluster systems are used in client/server configurations.

Chapter 7 presents high-integrity features of OpenVMS systems in distributed production system environments. It covers optional commercial-strength software that runs on OpenVMS, providing dependable transaction and database processing capabilities in open environments.

6 OpenVMS Systems in Distributed Environments

OpenVMS software is designed to support all computing environments, ranging from an environment comprising an OpenVMS standalone or VAXcluster system to a complex distributed environment involving software from multiple vendors.

This chapter describes integrated environments in which OpenVMS operating systems running optional (layered) software are connected to other systems provided by Digital and other vendors. The chapter summarizes software functions available in environments specific to OpenVMS and additional functions useful in open, distributed environments. It describes software running on OpenVMS that is relevant to distributed multivendor environments, such as networking products and NAS products that support multivendor application integration. The chapter also covers the participation of OpenVMS software in distributed client/server relationships, including the use of OpenVMS servers to provide services to clients on PCs and other desktop systems.

6.1 OpenVMS Functions Applicable to Distributed Environments

The OpenVMS system provides full computing capabilities to OpenVMS users in standalone, VAXcluster, and networked configurations (that involve connections to other OpenVMS systems and to other systems provided by Digital and other vendors).

A standalone OpenVMS system or a VAXcluster system that is not connected to other systems can support all basic software functions:

- OpenVMS base operating system functions (with built-in features such as help, Mail, and DECdtm and also integrated software such as VAXclusters, volume shadowing, and RMS Journaling)
- POSIX capability installed in the OpenVMS base operating system
- The full set of OpenVMS utilities
- Management of the OpenVMS system or VAXcluster system
- Full OpenVMS security features
- Program development capabilities
- The ability to run most OpenVMS layered products and applications

These computing capabilities, available to OpenVMS users on every standalone OpenVMS or VAXcluster system, are covered in Part II of this manual.

OpenVMS systems connected in a distributed multivendor environment perform the same functions as a standalone system but use the resources of the integrated environment. This type of environment can involve connection of different systems and integration of software from multiple vendors. Functions related to using OpenVMS software in open distributed environments include:

- Use of open standards-based software interfaces
- Use of portable software (such as POSIX)
- Use of networking connections
- Use of distributed applications
- Use of NAS software for multivendor interoperability
- Integrated management of open systems and multivendor enterprises
- Sharing of resources with other systems
- Participation in multivendor client/server configurations

These software features permit OpenVMS systems to function cooperatively in a network with other kinds of computers: Digital and other vendor systems, from PCs to supercomputers.

Open interfaces, portable software features, and distributed computing capabilities provided by the OpenVMS system are summarized in Chapter 2. The following sections describe the open, distributed software environments that OpenVMS systems support.

6.2 OpenVMS Systems in Distributed Heterogeneous Network Environments

Digital networking supports comprehensive distributed computing environments that permit OpenVMS and other Digital systems to be connected to systems supplied by other vendors. The networking software allows connected systems to share resources and function as an integrated system, providing basically the same environment for users everywhere in the network.

OpenVMS supports two basic types of networking services that can be integrated into a single multiprotocol network: the DECnet family of software and the DEC TCP/IP Services for OpenVMS (these networking products are introduced in Section 1.3.4). DECnet software (described in detail in Section 6.2.1) permits connection to virtually all Digital systems and, in its most recent phase, to all systems that support the international standards for Open Systems Interconnection (OSI). DECnet also provides gateways to other networking environments such as public packet-switching networks and the IBM SNA network. DEC TCP/IP Services for OpenVMS permit connection to UNIX systems on the Internet network (see Section 6.2.2). Supporting these networking services are products suitable for multivendor environments, such as multiprotocol wide area network (WAN) routers (see Section 6.2.4).

DECnet, OSI, and TCP/IP networks can be integrated to operate as a single network, in such a way that the protocols used are transparent to applications and users (see Figure 1–3).

6.2.1 DECnet Networking Software

The OpenVMS operating system participates in the DECnet network through its networking interface: DECnet software. In addition, optional products that run on the OpenVMS system provide networking functions that augment the DECnet capability.

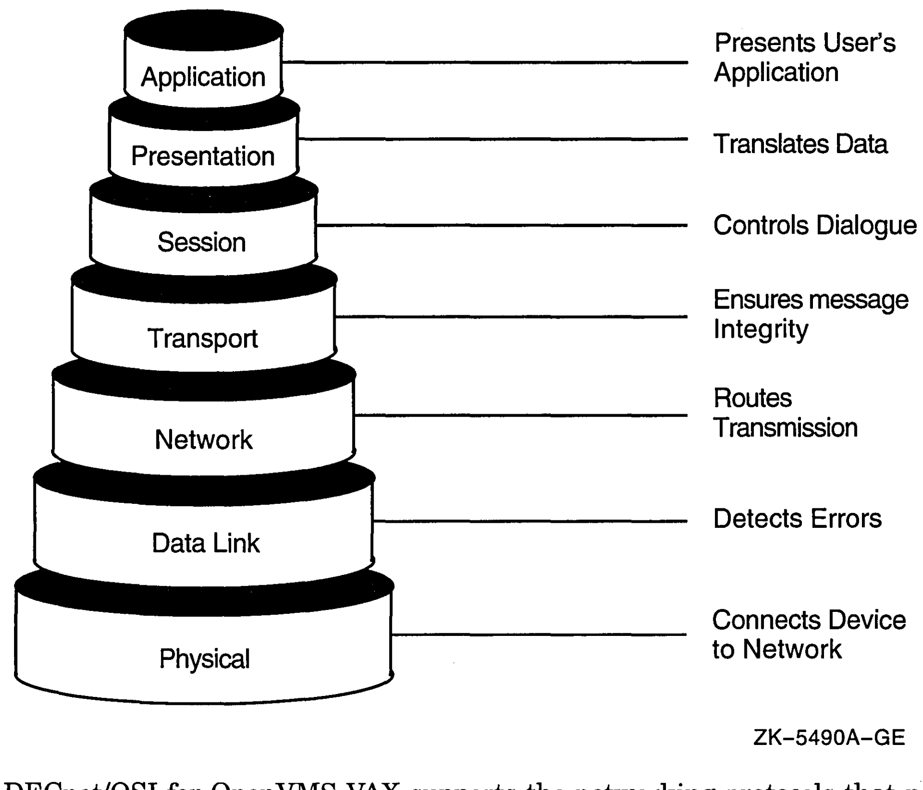
DECnet design is based on Digital Network Architecture (DNA) standards. Supporting the latest phase of DNA is DECnet/OSI, which conforms to international standards approved by the following standards organizations:

- International Organization for Standardization (ISO)
- International Telegraph and Telephone Consultative Committee (CCITT)
- Institute for Electrical and Electronic Engineers (IEEE)

DECnet/OSI supports the ISO standards defining the Open Systems Interconnection (OSI) model; these standards allow computer systems from different vendors to be interconnected. DECnet/OSI features a full OSI implementation with GOSIP certification.

Figure 6–1 shows the seven-layered OSI reference model that defines an environment for open networking. This model spans integration from the physical link to the application layer.

Figure 6–1 OSI Reference Model for Open Networking



DECnet/OSI for OpenVMS VAX supports the networking protocols that permit connection to other systems that use DECnet and to other vendor systems that support OSI. The existing DECnet for OpenVMS product (based on Phase IV, the previous phase of DNA) supports the protocols for DECnet communications.

Existing DECnet Phase IV systems, applications, and network components can continue to function in a DECnet/OSI environment. DECnet/OSI end systems can communicate with, and manage remotely, DECnet Phase IV nodes in the same network. DECnet/OSI end systems can transmit and receive messages, but cannot route messages through the network. Multiprotocol routers perform message routing in DECnet/OSI networks (see Section 6.2.4).

As implemented in DECnet/OSI for OpenVMS VAX, DECnet/OSI provides significant features that expand DECnet networking capability:

- Incorporates OSI capabilities to support open, multivendor networks.
- Provides support for very large networks. A network based on DECnet/OSI is, in effect, not limited in size. DECnet/OSI introduces increased addressing capabilities for large networks.
- Uses the Digital Distributed Name Service (DECdns), a networkwide name service that automates node name management, mapping node names to addresses. DECdns allows users to use network resources without knowing the network address of the resources. (Use of a networkwide namespace requires at least one DECdns server.)
- Uses the Digital Distributed Time Service (DECdts) to provide a precise, fault-tolerant clock synchronization for systems in a local and wide area network.
- Uses a network management architecture that defines manageable units as entities. Network management is modular and distributed, permitting remote management of all network functions.

DECnet/OSI for OpenVMS VAX supports the following software that conforms to ISO standards:

- OSI Transport Service software, which permits OSI applications to run on OpenVMS hosts
- VAX OSI Applications Kernel (OSAK) software, which implements the upper three layers of the OSI Reference Model
- File Transfer, Access, and Management Services (FTAM), which provide the ability to access and transfer files in an open systems environment
- DECnet/OSI for OpenVMS Virtual Terminal (VT) software, which permits remote logins and access to remote applications on any system that runs an ISO-compliant VT implementation

Other optional software products related to DECnet/OSI that can run on OpenVMS are listed in Table 6–1.

Table 6–1 OpenVMS Layered Products Related to DECnet/OSI

OpenVMS Layered Product	Function
VAX wide area network device drivers (WANDD)	Provide synchronous and asynchronous communication options support for DECnet/OSI for OpenVMS

(continued on next page)

Table 6–1 (Cont.) OpenVMS Layered Products Related to DECnet/OSI

OpenVMS Layered Product	Function
VAX P.S.I.	Implements the CCITT X.25 recommendations; lets users assign and use switched virtual circuits across a packet-switching data network
VAX P.S.I. Access	Allows DECnet software to access X.25 software on an X.25 multihost connector node
VAX OSI Applications Programming Interface (API) Toolkit	Provides a programming toolkit for developing OSI applications
VAX Message Router and VAX Message Router X.400 Gateway	Provides the ability to exchange mail in an open systems environment
Digital Management Control Center (DECmcc) Kernel	Provides kernel software for an integrated network management center capability

The Packetnet System Interface (P.S.I) software permits DECnet users to connect to packet-switching data networks (PSDNs) that conform to X.25 (such as TYMNET).

DECnet users can also access IBM SNA networks by means of Digital IBM Interconnect products, a family of networking products that include DECnet/SNA gateways and 3270 terminal emulators.

Techniques for network management are summarized in Section 2.4.2. DECnet/OSI network management is based on the director–entity model defined by Enterprise Management Architecture (EMA), described in Section 6.3.4. The entity-based network management architecture allows local and remote management of entities throughout the network.

The command line management interface for DECnet/OSI is the Network Control Language (NCL). NCL accesses management directives defined for DECnet/OSI entities using the Common Management Information Protocol (CMIP), based on ISO standards for encoding network management operations. NCL has the capability to perform basic management of all entities for a distributed system from a single location anywhere in the DECnet network. DECnet/OSI users can also use the DECnet Phase IV command line interface, NCP, to manage Phase IV nodes in the network.

Network management tasks for DECnet/OSI include the following:

- Management tasks:
  - Managing nodes in the local namespace and in the distributed namespace; managing the namespace
  - Reconfiguring network components using the NET$CONFIGURE command procedure
  - Converting Phase IV NCP commands to NCL commands
  - Setting up routing between DECnet Phase IV and DECnet/OSI areas
  - Downline loading, upline dumping, and controlling remote or unattended systems, using NCL and the Maintenance Operations Protocol (MOP)
  - Setting up network security, communications links, and the VAXcluster alias, using NCL

- Monitoring tasks:
  - Reporting network events during network operation, using NCL and the Event Dispatcher
  - Collecting information about the local network configuration
  - Displaying network status and characteristics, using NCL
- Problem solving tasks:
  - Testing network software and hardware, using NCL and Loopback tests
  - Testing network connections using DECnet Test Sender/DECnet Test Receiver (DTS/DTR)
  - Collecting and displaying information about specific protocol exchanges between systems in the network, using the Common Trace Facility (CTF)

OpenVMS users can perform the same general-user functions over the network that they can carry out for the local system, including remote file operations (see Section 5.5.3). They can issue DCL commands to perform operations over the network and develop command procedures and application programs to run over the network. In addition, they can send mail to remote nodes and use facilities like the Notes conference to share data. Examples of DECnet applications software used for general-user operations on the OpenVMS system are listed in Table 6–2.

Table 6–2 Examples of DECnet Applications for General-User Operation

DECnet Application	Function
File Access Listener	Allows heterogeneous file transfer over the network using the Data Access Protocol (DAP)
Network Virtual Terminal	Supports the ability to set host to another system, using the Communications Terminal Protocol (CTERM)
MAIL-11	Provides mail service over the DECnet network
VAX Notes	Allows computer conferencing

6.2.2 OpenVMS Connections to TCP/IP Networks

The DEC TCP/IP Services for OpenVMS product (formerly called the VMS/ULTRIX Connection product) promotes interoperability and resource sharing between OpenVMS systems and UNIX systems. This product makes OpenVMS a full participant in a TCP/IP network. It supports networking, file sharing, remote terminal access, electronic mail, and application development between OpenVMS systems and UNIX systems. Major components of the DEC TCP/IP Services for OpenVMS are listed in Table 6–3.

Table 6–3 Components of DEC TCP/IP Services for OpenVMS

Component	Description
Set of ARPANET communication system protocols	Supports industry-standard networking on OpenVMS through protocols based on the 4.3 Berkeley Software Distribution: TCP (Transmission Control Protocol) IP (Internet Protocol) FTP (File Transfer Protocol) Telnet Protocol UDP (UNIX Datagram Protocol) Other protocols (ARP, ICMP, RIP)
Remote Procedure Call	Allows application developers to partition applications along subroutine interfaces and have those subroutines execute on remote hosts
Network File System (NFS) server software	Permits UNIX clients to access OpenVMS files and files compatible with UNIX that are stored on OpenVMS
System management and DECwindows interfaces	Permits the OpenVMS system manager to manage Internet communications and the NFS server without detailed knowledge of UNIX networking

Users can connect OpenVMS, ULTRIX, and DEC OSF/1 systems with UNIX systems from other vendors, using industry-standard protocols for communication. Users can also write network applications that access the Internet protocols using standard OpenVMS services, including a QIO programming interface to access the lower level protocols.

DEC NFS for OpenVMS provides an NFS server on OpenVMS for UNIX clients. NFS is a protocol that gives network clients access to remote file services. This NFS server enhances data sharing among UNIX clients by providing a central data storage facility for OpenVMS and UNIX files. The UNIX client can access OpenVMS files or files compatible with UNIX that are stored on the OpenVMS server (which can be either an OpenVMS system or a VAXcluster system).

6.2.3 Network Security

OpenVMS security features protect DECnet nodes, user accounts, and data through password protection and file protection mechanisms. In general, the system manager can control access to a system through the use of proxy accounts and default accounts. The system manager can use the AUTHORIZE command to create proxy accounts, which specify remote users or groups of users who can access data on the local system with the same privileges as users logged in to the local system. For DECnet for OpenVMS (Phase IV), the establishment of logical links with remote nodes can be controlled through specification in the network database of the allowed logical link connections.

To help protect the security of messages transmitted over a local area network, the Ethernet Enhanced Security System can selectively encrypt information across a network, without affecting other network nodes not requiring encryption. This system includes the Digital Ethernet Secure Network Controller (DESNC) and the VAX Key Distribution Center (KDC) system attached to the LAN. The VAX KDC system provides tools to permit a system security manager to implement and manage a security-enhanced LAN. The DESNC and VAX KDC system implements a default access control policy that allows all nodes on a LAN network to communicate with each other, after successful authentication.

6.2.4 Other Supported Networking Protocols and Products

On local area networks, OpenVMS supports the following protocols:

- The LAT protocol, a highly efficient local area transport service that allows for connections to Digital and nonhost systems. The LAT protocol is used to communicate with other nodes and with devices (terminals, modems, or printers) offered by terminal servers on a local area network.
- The local area disk (LAD) protocol, used to access compact disc media that reside on a Digital InfoServer system.
- The local area systems transport (LAST) protocol, used for virtual disk access.

OpenVMS systems can use the LAD/LAST protocol to access a library of OpenVMS software and online documentation stored on CD–ROMs on the Infoserver. The CD–ROM (compact disc read-only memory) is an optical data storage technology formatted according to the ISO 9660 standard. The Infoserver is a dedicated hardware and software device that supports locally attached CD–ROMs and provides shared read-only access to CD–ROM discs. The InfoServer also provides support for Initial System Loading (ISL) via the LAN, permitting the OpenVMS operating system to be loaded from a compact disc.

OpenVMS hosts and other hosts can be connected to terminals over the LAN by means of terminal servers (the DECserver series). For example, a multiprotocol terminal server offers high-speed connections for terminals, printers, PCs, and other devices to any network service that supports TCP/IP. This DECserver can be used in OpenVMS, UNIX, ULTRIX, and DOS environments or in a network that combines these operating systems. This terminal server can be directly managed in multivendor network environments as a standalone unit or in a DEChub configuration. The server provides multivendor connectivity and wide area routable terminal service.

Multiprotocol wide area networking routers support routing in multivendor networks. The DEC WANrouter and the DEC Network Integration Server (DECNIS) are multiprotocol routers that provide a single routing service for DECnet/OSI nodes, DECnet Phase IV nodes, TCP/IP nodes, and OSI end systems. Some of these routers also implement X.25 routing circuits. The WANrouter and DECNIS use the Integrated IS-IS (Intermediate System-to-Intermediate System) routing protocol that conforms to the OSI model.

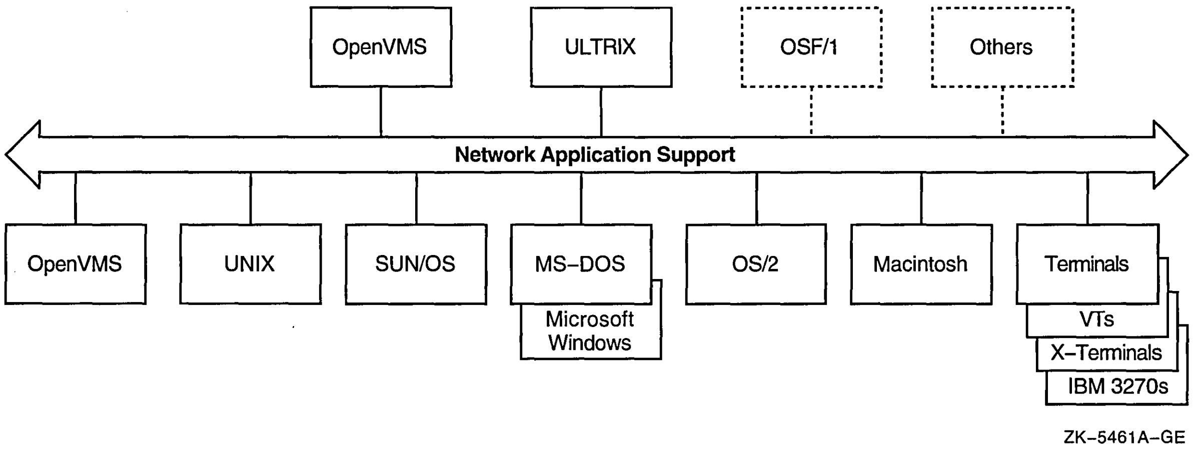
6.3 Multivendor Integration Using NAS Software

Network Application Support (NAS) software products, which conform to open systems standards, support multivendor computing capabilities in a distributed networking environment. NAS products run on OpenVMS systems and on systems from multiple vendors, promoting multivendor interoperability (as discussed in Section 2.1.5.2) and providing services for distributed applications (as described in Section 2.2.3).

NAS products and tools help applications from different vendors work together. NAS facilitates communication between applications so that different units of an enterprise or business can share information quickly and easily, even though the units may be physically distributed and may use different computer systems. The services supplied by NAS help make it possible for a distributed multivendor environment to appear to the user as an integrated environment.

NAS services are available for OpenVMS and many other systems from multiple vendors. Current NAS support for multivendor platforms is shown in Figure 6–2.

Figure 6–2 NAS Platforms



6.3.1 NAS Services

Application developers can use NAS services to create applications that provide transparent access to resources throughout a heterogeneous network. These applications are accessible on all NAS platforms.

To facilitate integration among different systems, NAS uses a client/server model of computing combined with application programming interfaces (APIs) and function-specific services. For example, an application (a client) requests information from another application (a server). The request is made through the API. The API communicates with the underlying NAS service, which executes the client's request. The process occurs transparently even though the client and server are on different systems on the network. The APIs, which pass data and control between client and server, implement industry and international standards, making the applications less dependent on any specific platform.

Six categories of modular NAS services work with APIs to transparently complete dialogues between the application and the elements it needs to address. Table 6–4 lists the categories of NAS services.

Table 6–4 NAS Services

Dialogue	NAS Services
User	Presentation Services
	Enable the application to interact with users. Some services operate across a variety of interface devices, while others depend upon a particular class of interface device.

(continued on next page)

Table 6–4 (Cont.) NAS Services

Dialogue	NAS Services
Applications	Communication Services Enable the application to communicate with other applications, both local and remote. They include electronic mail at the high level, and basic support for various communication paradigms at the low level (for example, remote procedure call and message queuing). Control Services Permit the application to control distributed program execution. They also permit application components to be effectively distributed across networked systems and then rapidly located at program execution time. Also included are services to invoke functions provided by application programs, making for a highly extensible system.
Data	Information Services Enable the application to define, store, access, and manipulate data. They provide a shared repository of information for tools and applications and for individuals and organizations within an enterprise. Multiple, distributed information models are supported.
System	Computation Services Enable the application to perform complex computations that operate on a variety of in-memory data. They support such functions as data manipulation, type conversion, international character handling, and time manipulation and synchronization. Management Services Permit the application to be easily and consistently managed from remote nodes. The same scheme is used to manage applications as is used for all of the information system components. Thus, system managers can manage anything from anywhere in a consistent fashion. The management services cover configuration, fault, performance, security, and accounting management.

Table 6–5 describes the NAS services currently available, associates the service with specific Digital products that can provide the service, and indicates the types of standards each service supports. All of the Digital products listed can also run independently on the OpenVMS operating system.

Table 6–5 NAS Services with Corresponding Digital Products and Standards Supported

NAS Services	Digital Products	Types of Standards Supported
Presentation Services		
Windowing services	DECwindows Motif	X Window System (endorsed by X/Open); OSF/Motif
Forms services	DECforms	FIMS (ISO)
Graphics services	DEC GKS, DEC GKS–3D, DEC PHIGS	GKS, GKS–3D, and PHIGS; ISO; ANSI; X Window System

(continued on next page)

Table 6–5 (Cont.) NAS Services with Corresponding Digital Products and Standards Supported

NAS Services	Digital Products	Types of Standards Supported
Presentation Services		
Terminal services	Terminal emulation products	ANSI
Information linking and navigation services	LinkWorks	No industry standards
Printing services	DECprint family of products	DPA; PostScript (de facto); ECMA; ISO
Communication Services		
X.400 mail transport services	MAILbus family of products, MAILworks	CCITT X.400
Message queuing service	DECmessageQ	OSF DCE; OSI; TCP/IP
Electronic data interchange services	DEC/EDI	ANSI; X.400
Remote procedure call services	DECrpc[1]	UDP/IP; OSF DCE
Control Services		
Application control services	ACA Services	OMG standards in process
Transaction management services	DECdtm	POSIX 1003.4a; OSF DCE
Thread management services	DECthreads[1]	OSF DCE
Information Services		
Compound document services	CDA family of products, DECimage Application Services	ODA/ODIF; ISO; SGML (ANSI/ISO); Abstract Syntax Notation (ASN.1, ISO); PostScript (de facto); X.400; CCITT; ISO Latin 1; ANSI and ISO color models
Data access services	DEC Rdb (including SQL and SQL/Services), RdbAccess family of products	SQL standards (ISO/ANSI)
Directory service	CDS[1]	ISO, OSF DCE in process
Repository services	CDD/Repository	ATIS
File sharing services	PATHWORKS family of products, DEC TCP/IP (including NFS)	NFS (de facto); IEEE; OSI; LAN Manager; MS–DOS; MS–NET; NETBIOS; TCP/IP; Telnet; FTP; UDP; ICMP, BIND; RIP
File transfer, access, and management services	FTAM products	ISO, GOSIP
Computation Services		
Distributed time service	DECdts[1]; DCE DTS; DECnet DTS	ISO; OSF DCE

[1]A component of the OSF DCE

(continued on next page)

Table 6–5 (Cont.) NAS Services with Corresponding Digital Products and Standards Supported

NAS Services	Digital Products	Types of Standards Supported
Management Services		
Management agent services	EMA Common Agent	CMIP; SNMP; OSF DCE
Management director services	DECmcc	ISO (CMIS and CMIP); ANSI; IEEE 802.2 and ISO 8802.2; OSF DCE
System Interface		
System interface services	POSIX for OpenVMS	ANSI/IEEE; ISO

6.3.2 Distributed Computing Environment Software

The OSF Distributed Computing Environment (DCE) (described in Section 2.2.4) is an integrated set of software services that supports the development, use, and maintenance of distributed applications. DCE facilitates the development of distributed client/server applications by making the underlying transports and systems transparent to application developers.

DCE capabilities are currently being implemented on the OpenVMS system. The optional DCE Developer's Kit for OpenVMS provides users with a subset of the distributed computing capability specified for the OSF DCE. Application programmers can use the Developer's Kit to begin the development and deployment of small-scale distributed applications in a heterogeneous environment prior to the availability of the full OSF DCE.

The following capabilities are included in the DCE Developer's Kit for OpenVMS:

- DCE Remote Procedure Call (RPC): Used to create and execute client/server applications; includes an RPC run-time library and IDL (Interface Definition Language) RPC stub compiler. It also includes a low-cost-of-entry local directory service (LDS), which is used to create DCE NSI-compliant client/server bindings, and an NSI proxy agent for PCs that provides a remote name service interface for PC clients.
- DCE Threads Service: Provides user-context multiprocessing capability; supported by the DECthreads package (which is also available on OpenVMS as described in Section 4.3).
- Documentation and sample applications for developing RPC applications.

The DCE components provided by this kit are interoperable with the corresponding components of the DEC DCE Starter Kits for the ULTRIX and DEC OSF/1 systems.

Other DCE technologies specified by the OSF DCE architecture are being implemented on OpenVMS, including the Distributed Time Service (DTS); the Cell Directory Service (CDS), which is based on DECdns; and DCE's distributed security service.

6.3.3 Application Development in Multivendor Environments

COHESION, the comprehensive CASE (computer-aided software engineering) environment, is a set of software development solutions, based on standards, that enable software development, reengineering, deployment, and management on a variety of computing platforms. The COHESION environment is designed to bring the open systems benefits of flexibility, investment enhancement, and multivendor interoperability and portability to software development.

The COHESION environment permits developers to build applications in the OpenVMS, ULTRIX, or PC environment for deployment across a network of systems from different vendors. COHESION is built on standards-based NAS services, enabling development on a full range of platforms. COHESION is also used to develop software for NAS environments: creating integrated applications with functions that can be distributed across a network of platforms from multiple vendors.

COHESION provides a variety of tools, developed by Digital and by other vendors, to support the entire software development life cycle. It integrates tools from multiple sources into a single development environment through a framework of the following levels of integration:

- Presentation integration: DECwindows Motif provides a common look and feel across tools. It allows programmers to interact with different tools in the same way.
- Control integration: DEC ACA Services (defined in Table 6–5) manage the flow from one activity to another, providing dynamic, functional integration. DEC ACA Services integrate functions from one tool to the next, permitting users to move from one activity to another in a natural manner.
- Data integration: CDD/Repository enables developers and applications to create, store, and access common definitions for data, objects, and methods. CDD/Repository is a distributed, active repository that connects the CASE, transaction processing, and end-user computing environments and ties together multiple tools from multiple vendors in the COHESION environment. It provides a single point of control for defining and sharing access to software development information.

COHESION provides integrated programming environments on the OpenVMS system. These environments provide powerful, flexible, easy-to-use toolsets to help software developers code as rapidly as possible, with a minimum of error. Examples of these tools are DECdesign, DECplan, and DECset.

DECdesign provides a DECwindows graphics environment for the analysis and design phases of the software development life cycle. It supports process, behavioral modeling, and data modeling with integrated techniques. Reports produced conform to CDA (Compound Document Architecture) and can be edited in other products that comply with CDA, such as DECwrite.

DECplan is an integrated time and project management tool designed to help users plan, track, schedule, and report on projects and meetings. DECplan is a client/server tool based on DECwindows, useful in large, heterogeneous networks.

The COHESION environment provides several DECset software tools that support software coding, testing, and maintenance of applications and data on OpenVMS systems and ULTRIX and DEC OSF/1 systems. The DECset tools include:

- DEC Language-Sensitive Editor/Source Code Analyzer (DEC LSE/SCA), used to develop and debug source code modules for applications. The LSE Editor has knowledge of the syntax of Digital programming languages and can provide context-sensitive help. The SCA allows interactive inquiries about the structure of the program.
- VAX DEC/Code Management System (VAX DEC/CMS), which tracks everything that happens to project files during development, including revisions of source code, documentation, data files, test files, system build descriptions, and requirements documents.
- DEC Test Manager, which simplifies the testing process by automating the organization and execution of tests.
- VAX DEC/Module Management System (VAX DEC/MMS), which automatically rebuilds the system after programmers make changes to application code.
- DEC Performance and Coverage Analyzer (DEC PCA), which collects and analyzes performance and test information; used to identify possible performance problems in an application.

The COHESION environment supports key programming languages and compilers, described in Section 4.2.2.

6.3.4 Managing Enterprisewide Multivendor Environments

The Digital Enterprise Management Architecture (EMA) extends the comprehensive approach to network management used by Digital networks to other vendors, networks, systems, and technologies. EMA is an open, extensible, object-oriented architecture that integrates all management functions, such as configuration changes, performance, and security. EMA provides for effective management of evolving, heterogeneous, multiprotocol, multivendor enterprises.

Network management for an enterprise is defined in the framework of EMA (see Section 2.4.3). EMA provides models for integrated management systems and self-managing network and system components, and lays out the director/entity framework for implementing automated management functions. The two major components of EMA are the manageable units called entities and the directors that manage the entities.

6.3.4.1 POLYCENTER Solution Software

The POLYCENTER solution for enterprise management is built on NAS and EMA. The basic POLYCENTER option includes software packages that provide base-level management capabilities in scheduling, fault and problem management, security, and storage management. Advanced network management applications perform higher level tasks such as integrating network information, loading it into databases, and presenting it to managers in graphic form. One of the POLYCENTER products is DECmcc.

6.3.4.2 DECmcc Software

The Digital Management Control Center (DECmcc), which implements EMA, is a multiplatform software system that can be extended to manage practically any entity across a distributed environment. DECmcc integrates network management software used to monitor, control, and test entities in any network, extending from a small homogeneous LAN to an enterprisewide heterogeneous distributed environment that includes components supplied by vendors other than Digital.

The DECmcc kernel is available with the DECnet/OSI for OpenVMS product (see Table 6–1). DECmcc can be used to manage the enterprise from a single site, such as an OpenVMS system.

6.4 OpenVMS Software in Multivendor Client/Server Environments

Client/server configurations provide a means of making a distributed, multivendor environment more effective. The client/server style of distributed computing (as described in Section 2.2.1) is highly flexible and adapable to a wide variety of software and hardware configurations. Servers make it possible to share resources, use data and applications on different vendors' systems, and provide the power of large systems to small client computers. Client systems can be integrated into enterprisewide networking systems, obtaining access to the resources of an enterprise while still retaining their own individual computing environments.

The following sections describe how OpenVMS client/server software is used in VAXclusters and how an OpenVMS system functions as a server to PCs in a PATHWORKS multivendor environment.

6.4.1 VAXcluster Servers and OpenVMS Clients

In any VAXcluster system, users can share computing, disk and tape storage, and batch and print job processing resources. Any node in the cluster can use clusterwide batch and print queues. VAXcluster technology also allows cooperating OpenVMS systems to share file and print resources over a LAN. This capability provides a mechanism for offering print and computing services to the network.

In a VAXcluster configuration, the mass storage control protocol server and the tape mass storage control protocol server make locally connected disks and tapes available across the cluster. A VAXcluster system can serve files and databases to the LAN for use by cluster members. In addition, databases can be offered over a VAXcluster using Rdb/VMS with SQL Services. These services enable a client OpenVMS system to issue SQL requests to a VAXcluster system that serves its Rdb/VMS databases to the LAN. The Rdb/VMS request is processed on the database server and the resulting information is sent to the SQL Services client.

Another client/server relationship on a VAXcluster involves DECwindows Motif display services. These display services offer a consistent interface for sharing display resources across a VAXcluster system. Users can display the output of programs running on any system in the cluster or any other system in the network.

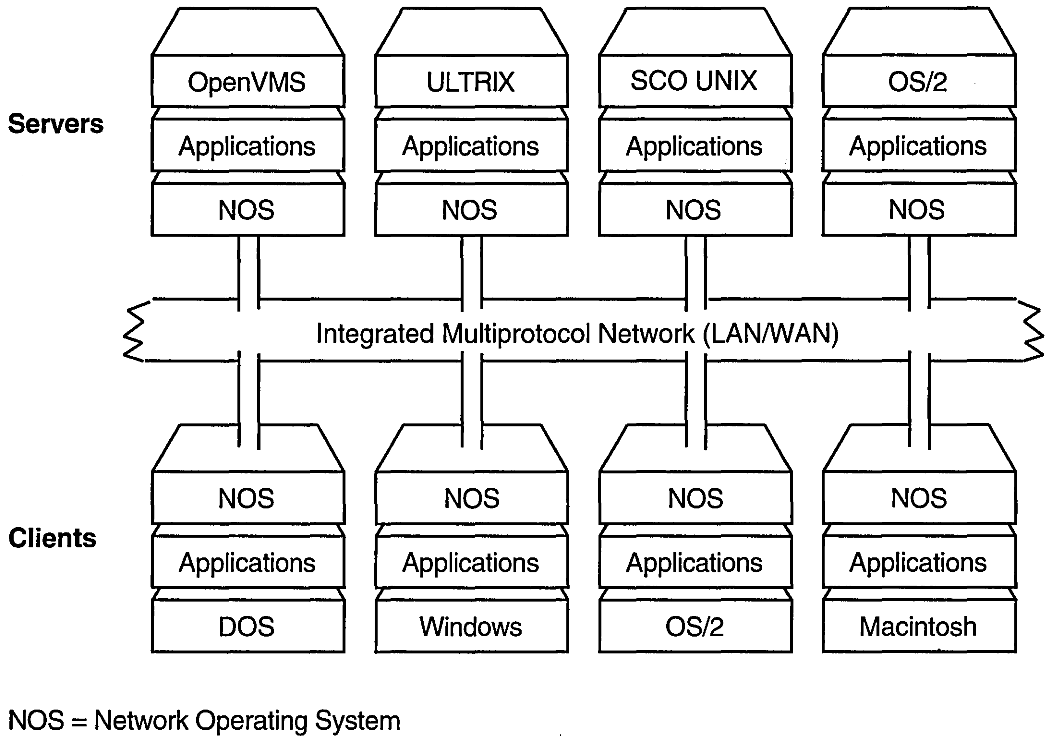
VAXcluster systems can also act as servers to clients on other vendors' systems in PATHWORKS configurations (as described in Section 6.4.2).

6.4.2 OpenVMS Servers in PATHWORKS Environments

PATHWORKS software permits the creation of an integrated desktop environment in which PC users can share information and resources in a LAN and access applications, data, and resources on OpenVMS servers and other servers across local and wide area networks.

The PATHWORKS family of network operating system software includes client software and server software: PATHWORKS client software running on DOS, Windows, OS/2, and Macintosh systems, and PATHWORKS server software running on OpenVMS, ULTRIX, SCO UNIX, and OS/2 operating systems (see Figure 6–3). To function as a server to PC clients, the OpenVMS system uses PATHWORKS for OpenVMS software in conjunction with the appropriate networking software products, including DECnet for OpenVMS and DEC TCP/IP Services for OpenVMS.

Figure 6–3 PATHWORKS Environment



Clients located on LANs use various networking transport protocols to communicate with servers. PATHWORKS network operating system (NOS) technologies provide basic network services.

PATHWORKS offers a choice of servers, clients, LAN technologies, network transports, and network operating systems, as follows:

- Servers include OpenVMS operating systems (running on all of the VAX processors) and ULTRIX, SCO UNIX, and OS/2 operating systems.

- Clients include MS–DOS (which can be running Windows), OS/2, and Macintosh systems.
- LAN physical media types include Ethernet and token ring.
- Network transports include DECnet, TCP/IP, IPX, NetBEUI, and AppleTalk, and access to X.25 and SNA via gateways.
- Network operating system technologies include basic network services for LAN Manager, NetWare, and AppleShare.

PATHWORKS client/server products are listed in Table 6–6. Services provided by OpenVMS servers to PC clients are discussed in Section 6.4.3. Networking connections between PATHWORKS clients and servers are described in Section 6.4.4.

Table 6–6 PATHWORKS Software for Clients and Servers

PATHWORKS Software	Client/Server Function	Description
PATHWORKS for OpenVMS	OpenVMS server	Acts as a file, disk, print, and mail server to DOS, Windows, or OS/2 clients over DECnet or TCP/IP connections.
PATHWORKS for OpenVMS (Macintosh)	OpenVMS server and Macintosh client	OpenVMS server acts as file, print, mail, and database server to Macintosh clients. Client and server share information and resources using AppleTalk and DECnet or TCP/IP connections.
PATHWORKS for DOS	DOS client	Uses facilities provided by OpenVMS, ULTRIX, SCO UNIX, and OS/2 servers over DECnet connections.
PATHWORKS for DOS (NetWare Coexistence option)	DOS client	Enables clients to connect to PATHWORKS servers over DECnet or TCP/IP connections, while concurrently using the facilities provided by Novell NetWare client software over IPX connections.
PATHWORKS for DOS (TCP/IP option)	DOS client	Enables DOS PCs to access OpenVMS, ULTRIX, and SCO UNIX server facilities over TCP/IP connections (contains TCP/IP networking software and utilities).
PATHWORKS for OS/2	OS/2 client and server	As client, uses PATHWORKS file and print services. As server, makes OS/2 file and print services available to other PCs over DECnet or TCP/IP connections.
PATHWORKS for OS/2 (TCP/IP option)	OS/2 client	Enables OS/2 client to access OpenVMS, ULTRIX, and SCO UNIX servers over TCP/IP connections.
PATHWORKS for SCO UNIX	UNIX server	Acts as a file, print, and mail server to PC clients over DECnet or TCP/IP connections.
PATHWORKS for ULTRIX	UNIX server	Acts as a file, print, and mail server to PC clients over DECnet or TCP/IP connections.

PATHWORKS configurations are flexible and can be changed easily. In most cases, PATHWORKS client software is originally stored on the server and loaded downline over the network to the PC when the PC is booted. The PC user can connect to a particular PATHWORKS server. PC clients can be connected to multiple servers at the same time.

Configuration changes, such as the addition of new PATHWORKS clients and servers, do not cause disruption to the PC user environment. PC users can continue to use the applications they prefer.

6.4.3 OpenVMS Server Capabilities for PATHWORKS Clients

Individual OpenVMS VAX systems and VAXcluster systems can provide services to PCs. An OpenVMS system running PATHWORKS for OpenVMS software can act as a server for personal computers that run DOS and OS/2 operating systems. An OpenVMS system running PATHWORKS for OpenVMS (Macintosh) software can act as a server for Macintosh systems. Both PATHWORKS products can exist on an OpenVMS system at the same time.

In a PATHWORKS client/server configuration, OpenVMS servers can be required to handle multiple requests coming from many systems at different times, and must handle requests quickly to ensure good response time at the personal computer or worksystem. In turn, the desktop systems must be served with the proper security and system management protection. Servers may offer data of a sensitive nature (for example, an OpenVMS server may provide financial spreadsheet data to an MS–DOS PC).

The following characteristics of the OpenVMS system make it an effective server for desktop PCs:

- OpenVMS is a multiuser system and resource server.
- OpenVMS is a multiprogramming, multiprocessing system with preemptive priority scheduling (as described in Section 3.1.1) and is capable of responding rapidly to many requests from different clients.
- OpenVMS is enhanced with security and system management tools and techniques based on years of supporting timesharing users.
- OpenVMS resource serving over the network is a natural evolution from resource serving to multiple users logged in to the system in a timesharing environment.

Using PATHWORKS to link PC clients with OpenVMS servers over a LAN permits the PC users to share files, applications, printers, and electronic mail. Additional functionality available to PC users includes database and transaction processing capabilities, terminal emulation, and X Window services. PC users can also take advantage of wide area networking capabilities, system management, and security features provided by the OpenVMS server.

Specifically, OpenVMS servers running PATHWORKS software can provide the following services (as illustrated in Figure 2–1) to PC and Macintosh clients running the appropriate PATHWORKS software:

- File services
- Disk services
- Print services
- Electronic mail services
- Database services
- Transaction processing
- Terminal emulation capability
- X Window services
- Management and resource control

6.4.3.1 File, Disk, and Print Services for PC Users

With appropriate PATHWORKS software, PC and Macintosh clients can access and store files on OpenVMS and VAXcluster systems anywhere in the local or wide area network. The remote file service appears to the PC client as a transparent extension of the PC's local disk. The PC files are automatically stored on an OpenVMS server in OpenVMS file format and can be shared transparently with users on DOS, OS/2, and Macintosh systems. Applications stored in files on OpenVMS servers can be run on the client, using the resources of the client. OpenVMS servers provide large system file capacity, data integrity, and security to PC files. The OpenVMS file services for PC clients are based on Microsoft LAN Manager and, for Macintosh clients, on the VAXshare Manager utility.

PATHWORKS disk services give DOS and OS/2 clients access to PC-formatted virtual disks located on OpenVMS VAX servers in local area networks. One or more clients can access a virtual disk on a server. With read-only access, many clients can copy and execute files (for example, DOS application files, utilities, and tools). Alternatively, with read-write access, only one client can write to a file (for example, a personal file). The disk services capability is often used to perform remote booting of DOS systems.

OpenVMS servers provide print services to PC clients using standard OpenVMS print queues. PC users can print files on a local printer connected to the PC or a remote printer connected to the OpenVMS server on a PATHWORKS network. PATHWORKS for OpenVMS (Macintosh) provides VAXshare remote print services to Macintosh clients.

6.4.3.2 Mail Services for PC Users

PATHWORKS mail services provide full OpenVMS mail handling capabilities to PC users. Using MAIL, the PC user can communicate with other PC users on the LAN and with PC and non-PC users throughout the enterprise. OpenVMS mail services support distribution lists, real-time mail notification, and other mail functions.

DOS PC users can also access DEC MAILworks (formerly known as ALL–IN–1 MAIL) on OpenVMS servers. DEC MAILworks conforms to the ISO/CCITT X.400 user agent standard and permits the user to exchange mail with users of any public or private mail systems that comply with the X.400 standard for message exchange throughout the world.

Users of basic MAIL and DEC MAILworks can exchange messages with X.400 messaging systems such as MCI Mail, facsimile systems, and proprietary mail systems such as IBM PROFS, as well as with ULTRIX systems.

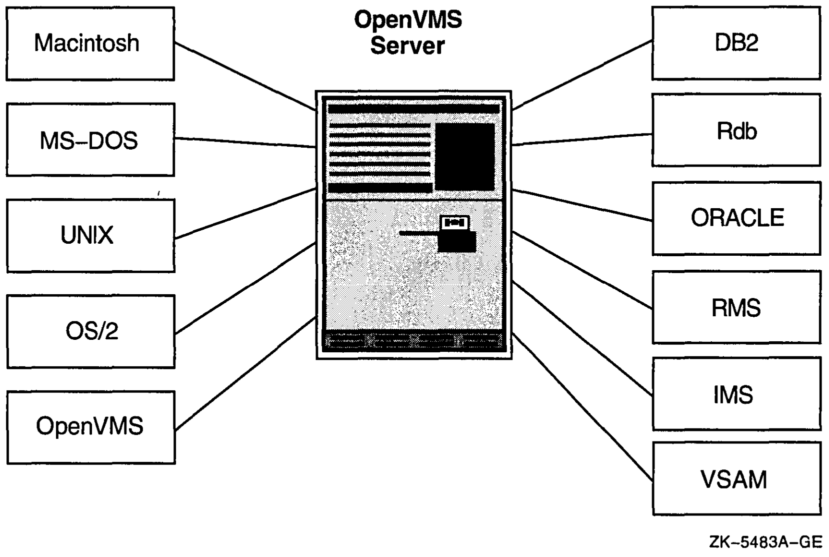
6.4.3.3 Database Services for PC Users

PATHWORKS can use data access services to give PC users transparent access to remote databases on OpenVMS and other servers. Database services based on SQL requests allow PC users to retrieve corporate data and use the input in PC applications. Examples of large databases on OpenVMS systems that PCs can access are DEC Rdb and RMS databases.

The Digital ACCESSWORKS products can be used to provide preconfigured client/server solutions based on the proven commercial strength of OpenVMS systems. DEC ACCESSWORKS, as illustrated in Figure 6–4, supports a range of client devices, including OpenVMS, UNIX, MS–DOS, OS/2, and Macintosh devices. It offers users concurrent, transparent access to a variety of multivendor data sources. DEC ACCESSWORKS supports the use of SQL for access to common databases such as DEC Rdb, RMS, ORACLE, and IBM DB2, VSAM, and

IMS databases. The various ACCESSWORKS solutions provide core software tools, NAS software, and server software, including PATHWORKS for OpenVMS network operating system software and DECnet and DEC TCP/IP Services software.

Figure 6–4 DEC ACCESSWORKS



See Section 7.3 for a discussion of database processing in multivendor environments.

6.4.3.4 Transaction Processing Capabilities for PC Users

PC users can act as full participating clients in transaction processing applications through the DECtp Desktop for ACMS product. The VAX ACMS (Application Control and Management System) transaction processing monitor running on OpenVMS is used to define, run, and control transaction processing applications (as described in Section 7.2.2). Desktop ACMS software enables PC, Macintosh, and OpenVMS clients to interact with ACMS applications on the OpenVMS server that accesses the database (for example, the DEC Rdb database). The clients establish networking connections to system servers using PATHWORKS software and DECnet, TCP/IP, or NetWare transports.

Desktop ACMS software links back-end application functionality to front-end forms management attributes such as displays, terminal characteristics, graphics, and color. ACMS applications can interact with many types of desktop computers and desktop user interface packages for Macintosh, MS–DOS, and Windows. Heterogeneous support enables installations to mix PC, Macintosh, and OpenVMS computers as front-end systems to the same ACMS application.

6.4.3.5 Terminal Emulation for PC Users

PATHWORKS client software permits a PC to access the host server system through terminal emulation (which enables a client to operate as though it were connected to the host computer by means of a terminal). For example, the PC user can access OpenVMS systems from the Windows interface and the DOS and OS/2 command lines, using VT320 terminal emulation. Windows allows the establishment of multiple terminal emulation sessions.

6.4.3.6 Windowing Services for PC Users

The X Window service provided with PATHWORKS for DOS is PC DECwindows Motif, which allows PC users to use X Window applications running on systems with an X client. For example, the PC can use the OpenVMS system running PATHWORKS server software to execute DECwindows Motif applications.

An optional PATHWORKS layered product that operates in the Microsoft Windows environment is the PATHWORKS eXcursion for Windows product. The eXcursion for Windows display server implements the X Window System protocol and makes use of DECnet connections. The eXcursion software visually integrates the X and Microsoft Windows environments; applications from both environments appear on a single screen and have the same appearance.

6.4.3.7 OpenVMS Management Services for PC Clients

OpenVMS servers offer additional services to PC users through PATHWORKS software:

- Security for PC data: The OpenVMS server provides password protection, access control, standard file protection, and security. Use of passwords permits control of access to network file, disk, and print resources. Passwords can be associated with file directories or printers being offered for use by OpenVMS servers. Security at the level provided by OpenVMS systems can be applied to PC files, including access control lists that define users' rights to access resources or services. Service and file access controls can be specified across a VAXcluster system, as well as on individual OpenVMS servers.
- Backup of PC hard disks: The OpenVMS server provides for unattended backup of PC hard disks through the use of timed batch files. During backup, PC file-access utilities prohibit unauthorized access.
- Management: The PSCA_MANAGER command line software provides PC users with command-line services for managing and controlling the PATHWORKS network, including control of client connections to the server.
- Menu-driven administrator's utilities: These utilities permit the system administrator to register clients, specify clients for remote booting, record network and server events, broadcast messages to PC users, and adjust server parameters to achieve efficient and reliable performance.
- Enterprise management using DECmcc: The DECmcc facility is a multiple LAN management facility that works with PATHWORKS. DECmcc is a NAS product; the DECmcc kernel is available with DECnet/OSI for OpenVMS (see Section 6.3.4.2). Through the use of DECmcc, the enterprise can be managed from a single site.
- Remote boot services for DOS PCs: The remote boot capability permits DOS PCs to activate their operating system, startup files, and PC networking software from the OpenVMS server on which the software is stored. Remote booting allows control of PC resources shared by networked PCs and facilitates system administration of multiple PCs.

6.4.4 PATHWORKS Network Connectivity

This section summarizes networking capabilities supported by PATHWORKS.

PATHWORKS provides PC users with a family of network integration products that supply multivendor LAN connectivity, permitting users on one LAN to communicate with users on another LAN. PC users can also connect from LANs to the rest of an enterprise through WAN network connections.

In PATHWORKS configurations, PC clients on LAN physical media use networking transport protocol software to communicate with OpenVMS and other servers. Network operating systems (NOSs) provide basic networking services required for PATHWORKS connections.

Each server and client node that runs PATHWORKS software uses a network transport. PATHWORKS supports a broad range of network transports, as listed in Table 6–7. OpenVMS PATHWORKS servers support DECnet, TCP/IP, AppleTalk, LAT, and LAST transports.

Table 6–7 Network Transports Supported by PATHWORKS Software

Transport	Description
DECnet	Used to provide file and printer services on LANs and WANs to all clients that connect to an OpenVMS server. DECnet gateways support connections to X.25 networks and IBM SNA networks.
TCP/IP	Used to provide file and printer services to DOS or OS/2 clients that have installed the PATHWORKS for DOS (TCP/IP) or PATHWORKS for OS/2 (TCP/IP) product. TCP/IP supports connections to Internet networks.
AppleTalk for OpenVMS	Implements AppleTalk protocols in the OpenVMS environment (provided by the PATHWORKS for OpenVMS (Macintosh) product); used by Macintosh clients for file and print sharing.
IPX	The Novell NetWare Internetwork Packet eXchange protocol (provided by the PATHWORKS for DOS [NetWare Coexistence option] product).
NetBEUI	The transport layer of the network basic I/O system (NETBIOS) protocol that lets OS/2 users connect to an IBM LAN server as well as to a PATHWORKS server. NetBEUI is also supported by the SCO UNIX server.
LAT	The Digital local area transport used for terminal communications and print services for DOS and OS/2 clients.
LAST	The Digital local area system transport used for disk and some file services for DOS and OS/2 clients.

Other PATHWORKS for OpenVMS (Macintosh) connections are as follows:

- AppleTalk-to-DECnet Gateway (OpenVMS software), which lets a Macintosh application connect to a DECnet service on any server.
- DECnet for Macintosh, which lets Macintosh computers participate as Phase IV end nodes in a DECnet network.
- Macintosh Communications Toolbox, which provides tools used by Macintosh applications for network connections, terminal emulation, and file transfers.

The following types of physical interconnect technology can be used in the LANs to which the PCs are connected:

- Ethernet, which provides connection between DOS, OS/2, and Macintosh clients and OpenVMS servers
- Token ring, which provides token ring network connection between DOS and OS/2 clients and OpenVMS servers equipped with TRNcontroller adapters
- LocalTalk, which provides connection of Macintosh clients to OpenVMS servers through gateways

Network operating system technologies include the file and print services for LAN Manager, NetWare, and AppleShare. Connectivity for DOS and OS/2 PCs is based on the LAN Manager from Microsoft Corporation, and the OpenVMS and ULTRIX servers are adaptations of LAN Manager sources as well. NetWare from Novell allows PCs to function simultaneously as PATHWORKS clients and NetWare workstations and permits users on NetWare LANs to use the PATHWORKS services available to DOS clients. The NOS for Macintosh systems is AppleShare (supplied by Apple Computer, Inc.); AppleShare requires that the AppleTalk network transport protocol be used as the file and print transport.

PATHWORKS also interoperates with other PC LANs, so PC users can concurrently access resources on both PATHWORKS and other vendors' NOSs.

7 OpenVMS Systems in Commercial Environments

OpenVMS systems incorporate commercial-strength software capabilities that make them highly suitable for use in critical production system computing environments and in other configurations that require dependable, high-integrity software.

This chapter describes the capabilities of OpenVMS systems as distributed production servers in open environments. It also covers OpenVMS support for transaction processing and database applications that can run in distributed multivendor environments.

7.1 OpenVMS Production Systems

OpenVMS systems provide high-integrity, commercial-strength capabilities that meet the needs of production systems. A production system is a computing system configuration essential or critical to the operation of an organization. Production systems can exist in a variety of computing environments, ranging for a single system, cluster, or client/server configuration to complex factory or commercial configurations involving desktop devices and mainframes.

A production system need not be limited to one system or cluster. A distributed production system can involve a collection of databases and processors, linked by high-speed, high-throughput networks, with enabling software that makes the configuration a single virtual database and processor to the end user, operator, and system manager.

OpenVMS systems incorporate powerful, dependable production system capabilities (as summarized in Section 2.3). In addition, OpenVMS systems support reliable distributed processing in integrated multivendor computing environments (as described in Chapter 6). OpenVMS systems can be used with optional software that runs on OpenVMS to meet the following needs of distributed production systems:

- The production system must remain up and running with little or no interruption.
- The system must be able to produce data, gathering information from the real world and processing the data.
- The system needs to present data easily on demand.
- The system must be able to handle heavy work loads.
- The configuration must not be limited in size.
- The system must be able to process a variety of applications.
- The system should not be limited to any particular field of activity.

- The system should be able to support transaction processing in open environments.
- The production system can be part of a global information system.
- Production systems can be at the core of task-driven solutions to overall large computing problems in organizations or departments.
- Production system environments can involve the use of PCs as terminals over PATHWORKS connections to OpenVMS servers (see Section 6.4.2).

7.1.1 OpenVMS Distributed Production Servers

The OpenVMS system provides the features needed for production systems of any size or configuration. OpenVMS configurations supply computing power, manageability, and security in predictable, stable environments. OpenVMS software ensures high application availability and data integrity. OpenVMS systems are dependable, exhibiting reliability, recoverability, and fault tolerance. These commercial-strength attributes of OpenVMS are summarized in Section 2.3. For detailed information about dependable systems, refer to *Building Dependable Systems: The OpenVMS VAX Approach.*

OpenVMS systems are extremely well suited to function as production system servers in multivendor configurations involving distributed processing of applications. OpenVMS client/server capabilities in distributed environments are described in Section 6.4.

OpenVMS production servers support distributed production systems that involve high-integrity, fault-tolerant, and transaction processing software. OpenVMS servers can be individual OpenVMS systems or VAXcluster systems that can coexist in a heterogeneous network with servers from other vendors (for example, IBM servers). Client software can reside on OpenVMS systems and on any other vendor's system that can run distributed applications. NAS client/server software services can facilitate integration of multivendor client/server configurations into a single distributed computing environment (see Section 6.3).

7.1.2 Providing Dependability in Production System Environments

OpenVMS meets the requirement of production systems for dependable software that ensures availability and data integrity. Dependability attributes of OpenVMS systems are discussed in Section 2.3.1. This section highlights the functions of particular OpenVMS integrated and optional software products that promote availability and data integrity in production systems.

Software for managing and monitoring production system environments is described in Section 7.1.3. Transaction processing and database management software that runs on OpenVMS systems is covered in Section 7.2 and Section 7.3.

7.1.2.1 Maintaining High Availability in Production System Environments

Software that guarantees high availability of OpenVMS systems in production system environments includes VAXclusters, volume shadowing, and the DEC Availability Manager for Distributed Systems (DECamds).

OpenVMS VAXcluster systems form an ideal base for developing and running high-availability applications, such as transaction processing applications and server applications. VAXcluster configurations provide for data sharing and independent failure characteristics (see Section 2.3.2). Some of the major features of VAXcluster systems that promote availability are:

- Ability to survive failure of any component

- Ability to be serviced without interruption to applications
- Functions as a single system but individual systems can leave the cluster while the cluster continues to operate
- Transparent sharing of resources and data
- Centralized data can be made available to large numbers of users
- Balanced work load across print and batch queues
- Can grow with an organization
- Flexible configurations, including large disaster-tolerant configurations (see Section 7.1.3.5)
- Support for the full range of OpenVMS systems on VAX processors

OpenVMS Volume Shadowing is a system integrated software product that provides for high data availability for disk storage devices (see Section 2.3.2). Volume shadowing involves maintaining the same data on multiple disks, called a shadow set. Volume shadowing provides these availability features:

- Prevents data loss resulting from media deterioration, communication path failure, or through controller or device failure
- Provides continued access to data despite failures in the disk media, disk drive, disk controller, or OpenVMS host serving a shadow set member
- Prevents storage subsystem component failures from disrupting system or application operations
- Continues operation without interruption if one volume fails, automatically directing requests for data to an alternate volume
- Supports controller-based and host-based shadowing, including shadowing of widely separated VAXcluster systems connected by LANs
- Usable on any Digital Storage Architecture (DSA) disk and Digital SCSI disks on any VAX processor or VAXcluster system

Another tool used by OpenVMS system managers to improve availability is DECamds, a real-time monitoring, diagnostic, and correction tool. It collects and analyzes data from multiple nodes simultaneously, directing output to a centralized DECwindows display. DECamds detects resource availability and suggests corrective actions.

To improve system availability, DECamds provides the following features:

- Alerts users to resource availability problems, suggests paths of investigation, and recommends actions to improve availability
- Allows real-time intervention, even when remote nodes are hung
- Centralizes management of remote nodes over the LAN
- Provides an intuitive DECwindows graphical user interface that is easy to learn and use

DECamds runs on a standalone console, gathering data from remote nodes. The application displays two windows: the event log window oriented toward corrective action, and the system overview window that offers user-directed data collection and display options.

7.1.2.2 Ensuring Data Integrity in Production Systems Environments

OpenVMS products that promote integrity on OpenVMS systems include RMS Journaling software and DECdtm services.

RMS Journaling software protects the data integrity of RMS files. RMS Journaling supports three types of journaling, each providing protection against a particular type of problem:

- After-image journaling that permits changes made to a data file to be redone: Designed to recover a file that was lost or corrupted through accidental deletion or device failure.
- Before-image journaling that returns a data file to the previous state, undoing changes made to the file: Useful if bad or erroneous data is entered into a file during an update session.
- Recovery-unit journaling that ensures a series of RMS operations called for by an application program are done in their entirety or not at all: Protects against inconsistent data due to the partial completion of transactions.

Recovery-unit journaling provides for transaction integrity. A transaction is a series of RMS record operations on one or more files and is viewed as an atomic operation, that is, all operations must be completed or none are (see Section 7.2).

The set of DECdtm services on the OpenVMS operating system (summarized in Section 3.1.6) provides basic operating system support for distributed transactions (as described in Section 7.2.3). DECdtm ensures transaction and database integrity during transaction processing through the use of the two-phase commit protocol, which ensures that all database resources commit to completing a transaction or none commit.

Database products running on OpenVMS that ensure data integrity include DEC Rdb (as described in Section 7.3.1.1) and DEC DBMS (as described in Section 7.3.2).

7.1.3 Managing and Monitoring Production System Environments

Production system work can be located in centralized data facilities or can be dispersed throughout a distributed environment. A distributed production system environment can involve a number of diverse systems used by a large number of people performing varied tasks. Managing complex computing environments often involves the use of multiple products and services.

OpenVMS supports a wide variety of optional software tools and utilities to assist users in managing complex distributed production system environments. Section 2.3.5 summarizes OpenVMS capabilities for managing and providing security for large datacenters and dispersed environments. Software running on the OpenVMS system that can be used in managing complex datacenters is listed in Table 7–1. The table covers optional software products and also the following software supplied with the OpenVMS operating system: Accounting utility, AUTOGEN, Error Log utility, generic queues, Monitor utility, OPCOM message routing, and RMS Journaling.

The following sections provide additional information about some of the tools running on OpenVMS that are used to manage and monitor distributed production system environments.

Table 7–1 Software Used to Manage Large Datacenters

Software Products	Function
	Accounting
OpenVMS Accounting	Facilitates user control
DECperformance Solution (DECps/AC)[1]	Provides basic user accounting and chargeback reports
	Application Management
VAX ACMS and DECtp Desktop for ACMS	Provide recoverable applications using a transaction processing monitor
DECdecision	Offers an easy-to-use method for collecting and categorizing data
DEC RALLY	Provides a user-friendly interface for transaction processing
VAX TEAMDATA	Allows data collection and analysis through an easy-to-use interface
	Application Debugging
DEC Performance and Coverage Analyzer (DEC PCA)	Tunes and checks applications
	Archiving
VAX Storage Library System (SLS)	Protects user data on other media
	Capacity Planning
DECperformance Solution (DECps/CP)[1]	Provides a capacity planning function to help the system manager analyze how changes in the configuration affect users' performance
	Event Messages
OPCOM message routing	Provides event notification
Error Log utility	Selectively reports the contents of an error log file containing events written by the OpenVMS system
VAXcluster Console System (VCS)	Consolidates system consoles and analyzes console messages for events
DECalert	Manages sophisticated event notification
POLYCENTER System Watchdog	Provides for automated monitoring, detection, and reporting of hardware and software events, including events defined by a user application
	Job Scheduling
DECscheduler	Schedules batch and print jobs
Generic queues	Uses VAXcluster queues to feed queues on specific members across the cluster

[1]Requires DECps/DC (Data Collector)

(continued on next page)

Table 7–1 (Cont.) Software Used to Manage Large Datacenters

Software Products	Function
	Data Access
DEC DBMS	Offers a CODASYL-compliant database with journaling, online backup, online database reconfiguration, and transaction processing support
DEC Rdb for OpenVMS	Offers a relational database with journaling, online backup, selected online database reconfiguration, and transaction processing support
DEC RdbAccess for ORACLE on OpenVMS	Provides a DEC Rdb SQL gateway, permitting direct, read-only access to ORACLE databases on the OpenVMS operating system
DEC RdbAccess for VAX RMS on OpenVMS	Provides a DEC Rdb SQL gateway for direct, read-only access to Digital RMS and IBM VSAM data sets
VIDA for DB2	Provides a DEC Rdb gateway, permitting direct, read-only access to IBM DB2 databases
RMS Journaling for OpenVMS	Provides data recovery and transaction processing support
	Disk Striping
VAX Disk Striping Driver	Spreads I/O load over multiple disk drives
	Forms Management
DECforms	Separates form input from application management
	Mirrored Disks
OpenVMS Volume Shadowing (Phase II)	Helps system survive disk failure
	Disk Defragmentation
DEC File Optimizer for OpenVMS	Defragments disks by using the OpenVMS movefile subfunction
	Performance Management
AUTOGEN command procedure	Optimizes system parameter settings based on usage
DECperformance Solution (DECps/PA)[1]	Identifies bottlenecks and recommends ways to fix problems
DECram (in-memory disk/device)	Alleviates I/O bottlenecks
Monitor utility	Provides basic performance data
	Remote System Management
DECmcc	Performs industry-standard network and system management system functions
VAXrsm (Remote System Manager)	Provides easy-to-use remote management of OpenVMS or ULTRIX systems

[1]Requires DECps/DC (Data Collector)

(continued on next page)

Table 7–1 (Cont.) Software Used to Manage Large Datacenters

Software Products	Function
	System Configuration
Remote Bridge Management System (RBMS)	Maintains a database of the setup of a system's complement of network servers and facilitates automatic loading of the servers upon system initialization
Terminal Server Manager (TSM)	Maintains a database of the setup of a system's complement of terminal servers and facilitates automatic loading of the servers upon system initialization
VAXsimPLUS	Monitors device status and alerts the system manager of impending failure

7.1.3.1 Storage Management Products

Optional products developed by the Digital Enterprise System Management (ESM) Program are used to manage mass storage devices and the data files stored on them in distributed production system environments. Examples of mass storage products that run on OpenVMS are discussed in this section.

Reducing file fragmentation on OpenVMS systems is possible with the DEC File Optimizer for OpenVMS. This product provides the ability to reduce file fragmentation, schedule defragmentation jobs, optimize file placement on the disk, and select or exclude files. If a user accesses a file that is being defragmented, the file will be released and defragmentation will continue on another file. The defragmentation engine reads the disk's file structure and dynamically determines which files should be placed at which locations on the disk. "Hot files" (those files on the disk that are accessed most frequently, determined using the DECps tool) can be moved to the center of the disk. Files are moved to different locations on the disk through invocation of the OpenVMS atomic movefile system service.

The VAX Disk Striping Driver for OpenVMS combines multiple disks into one virtual disk or stripeset. Each stripeset, composed of multiple disk drives, appears to applications and utilities as a single "pseudodevice" capable of responding to I/O operations defined for ordinary disk drives. Through the use of parallel I/O technology, the Disk Striping Driver can spread a file over several disk drives and read or write it over several channels simultaneously. Use of a stripeset increases I/O performance over the use of a single disk drive or a set of unstriped disks. This product solves I/O performance problems and is useful in load balancing on OpenVMS systems.

The VAX Storage Library System (VAX SLS) automatically schedules backup operations for VAXcluster systems, allowing the storage administrator to define which files should be backed up using what schedule. SLS submits the required backup jobs on schedule and tracks their progress. SLS automatically manages the usage, allocation, and tracking of magnetic tapes and write-once-read-many (WORM) optical disks.

7.1.3.2 DECram for OpenVMS Device Driver

DECram for OpenVMS is an integrated disk device driver that enables OpenVMS system managers to create pseudodisks in the VAX system main memory to improve I/O performance. The OpenVMS operating system can use standard OpenVMS disk I/O operations to access data on a DECram disk, but at a much greater access rate than for standard hardware disks. The DECram disk, resident in main memory, is accessed through the OpenVMS file system in the same way as physical disks are accessed; no changes are required to applications or system software. DECram disks may be used to store frequently needed read-only data (such as commonly used image files) or to hold temporary or scratch files for applications. The DECram disk information is volatile; it is not saved when the system shuts down or crashes.

7.1.3.3 Performance Management Tools

Optional software running on OpenVMS that provides assistance in managing performance includes the following tools:

- DECtrace collects and reports on event-based data. The event data can be gathered from any OpenVMS layered product or application that contains DECtrace service routine calls. Data can be used for performance analysis, application or database tuning, or error logging.
- RdbExpert assists administrators and designers in enhancing Rdb/VMS database design and performance and achieving optimal throughput for database-intensive applications.
- DECperformance Solution (DECps) provides capabilities for performance measurement and capacity management. DECps increases productivity by automating data collection, utilizing artificial intelligence (AI) techniques for performance-tuning recommendations, and enabling interactive modeling for capacity planning. Extensive graphing and reporting facilities supply essential details of system activity. The information provided by DECps helps to optimize the use of current resources as well as to plan more accurately for growth. DECps enables datacenter managers to use a more proactive approach, rather than a reactive approach, to planning.

  DECps is an integrated set of OpenVMS layered products, as follows:

  – The DECps Data Collector (DECps/DC) gathers and manages OpenVMS system data; it must be installed on each system.
  – The DECps Performance Advisor (DECps/PA) provides performance analysis, graphing, and reporting capabilities; it analyzes data collected using expert system techniques and generates a statistical overview of actual system performance.
  – The DECps Capacity Planner (DECps/CP) determines system performance for various work loads and configurations, using data collected to perform capacity planning exercises.
  – The DECps Accounting Chargeback (DECps/AC) allocates charges for resource usage and generates reports that can be used as an itemized bill or a general resource utilization report.

See also Section 7.1.2.1 for a description of the DEC Availability Manager for Distributed Systems (DECamds), which provides real-time observations of system resources.

7.1.3.4 Optional VAXcluster System Management Software

OpenVMS supports optional software products that provide management capabilities and services for complex distributed environments. These tools can run on VAXcluster systems.

OpenVMS can manage other systems remotely over the network. The VAX Remote System Manager (VAXrsm), which runs on OpenVMS, is an optional network product that permits a system manager to manage a number of OpenVMS systems and ULTRIX systems connected to a DECnet network. The system manager operates from a VAXrsm management server to install software remotely and schedule or initiate backups for the remote client's system. VAXrsm permits the manager to manage more systems and offloads system management tasks from users on individual systems. VAXrsm supports client systems ranging from single workstations to VAXcluster members to PC LAN servers.

VAXsimPLUS is a knowledge-based predictive maintenance tool that continually monitors all system devices to detect problems before they result in down time. VAXsimPLUS predicts impending failures and provides dynamic disk substitution and restoration across DECnet, VAXcluster, and other systems.

The VAXcluster Console System (VCS) consolidates the console management of distributed heterogeneous equipment. One VCS host can manage multiple computing resources over standard lines. VCS notifies personnel of critical systems events and/or activates corrective action routines.

DECalert consolidates system and user alarms and notifies the appropriate personnel. It runs on OpenVMS and provides client support for OpenVMS and ULTRIX systems. DECalert can significantly reduce down time by alerting the right people quickly, sending a variety of messages by means of pagers, electronic mail, and electronic voice (DECtalk).

DECscheduler for OpenVMS automatically schedules, executes, and monitors tasks such as backup, file maintenance, and production jobs. It also supports complicated, repetitive batch applications such as payroll, manufacturing resource planning, and financial consolidations.

The POLYCENTER System Watchdog Agent (WDA) and Consolidator (WDC) are optional products that provide automated detection, notification, and correction of system faults and problems. The System Watchdog monitors network, systems, subsystems, processes, and user-defined external events, detecting and reporting changes in resources before serious problems occur, in order to optimize system availability for users. The POLYCENTER System Watchdog (which replaces the Data Center Monitor [DCM] product) is part of the integrated set of system management tools provided by the POLYCENTER Solution (see Section 6.3.4.1).

7.1.3.5 VAXcluster Multi-Datacenter Facility

The optional VAXcluster Multi-Datacenter Facility (MDF) (described in Section 2.3.5) provides very high levels of system availability and data protection. The MDF enables management of widely distributed VAXclusters that are connected in disaster-tolerant configurations. MDF integrated software permits VAX CPUs in multiple datacenters to be combined into a single manageable VAXcluster system. The MDF consists of datacenters that are located a safe distance apart, connected using the high-speed FDDI fiber optic interconnect. The MDF can be managed from either location, even if failures occur at one of the datacenters. Site redundancy enables recovery from any single failure, such as power loss, to maintain application availability.

7.2 Transaction Processing in Multivendor Environments

Transaction processing applications are typically described as critical or "bet your business" applications, which consequently require the highest degree of availability, security, and integrity. These applications are usually high-volume, online applications that process transactions in real time and reflect the most current state of the business.

Transaction processing (TP) is the implementation of computer transactions, which supply the mechanism for accomplishing a business transaction. A computer transaction is a series of record operations made on one or more files. Often several computer transactions are required to implement a single business transaction (such as making a travel reservation or making an electronic funds transfer from one bank account to another).

The transaction application consists of one or more application programs that receive the input data and initiate the required transaction. Digital transaction processing software provides the underlying integrity for the application, monitoring the transaction through completion or, in the event of failure, canceling every step.

7.2.1 Distributed Transaction Processing Systems

Traditionally, transaction processing applications have been developed to run on a single computer such as a large mainframe system. The trend is toward distributed client/server transaction processing systems, in which an application runs on a set of systems linked in a network.

Transaction processing solutions provided by Digital permit transactions to be executed at multiple sites and combined with data stored over widely distributed nodes, or to be centralized on a single computer. Distributing transaction processing applications across multiple systems provides the following benefits:

- Local data can be kept close to users.
- Processing can be offloaded from more expensive mainframe systems.
- Higher overall throughput can be achieved.
- Communication costs can be cut.
- Scalable solutions can be configured to meet changing needs.

Transaction processing software that runs on the OpenVMS system involves TP system functions that are separated into discrete components with specified interfaces between them. Individual components can be changed without affecting the others and can be redistributed without impacting the applications. Users can choose to mix and match components from various vendors as long as the components comply with the standard interfaces.

High-level software systems available on OpenVMS systems provide capabilities needed for the core applications of commercial customers:

- DECtp is a transaction processing system that provides control and management of TP applications involving multiple users accessing a common database. The DECtp system is based on the ACMS transaction processing monitor (see Section 7.2.2), DECforms (see Section 5.4.1), and the DEC Rdb database management system (see Section 7.3.1), along with CDD/Repository and COHESION development tools. For multivendor desktop systems, the DECtp Desktop for ACMS product (described in Section 6.4.3.4) is available.

- DEC RTR (Reliable Transaction Router) is a distributed software message routing system that supports TP applications (see Section 7.2.4).

7.2.2 Distributed Transaction Processing Monitors

A transaction processing monitor is a collection of components bundled together in a tightly integrated package to facilitate application development and provide high performance and dependability in the production environment.

VAX ACMS is an application control and management system for developing, controlling, and maintaining transaction processing applications. It consists of front-end clients that collect multiple transactions for the back-end servers that execute the database I/O. ACMS provides operating system interfaces, standard menu-driven functions, terminal and file I/O services, and other commonly needed services for building transaction processing applications, using most OpenVMS higher level languages. It is designed for high-productivity application development.

VAX ACMS is a highly reliable TP system that includes dependability features to keep applications up and running at optimal efficiency, recoverability, and durability. Some of the major dependability features are:

- Queue management
- Load balancing and scheduling
- Application scheduling
- Deferred transaction scheduling
- Transaction routing
- Security
- Menu subsystems
- Message recovery
- Debugging support
- Interfaces to nonstandard or other-vendor devices

VAX ACMS is fully integrated with DEC Rdb, DEC DBMS, DECdtm services, DECforms, and the CDD/Repository. This group of products is part of the Digital COHESION environment for providing development, run-time, and management features for transaction processing applications. The high-availability features inherent in these software products combine to provide smooth transitions when problems or failures occur. Combining VAX ACMS with DEC Rdb database management (described in Section 7.3.1) improves the system's ability to resist most failures.

Features of ACMS that provide for fault management or recovery from failures that can occur during run time include:

- Automatic front-end terminal failover
- Use of queues to capture user requests
- Balancing of process pools
- ACMS application failover

For additional information about ACMS dependability features, refer to *Building Dependable Systems: The OpenVMS VAX Approach*.

VAX ACMS works with DECdtm services provided by the OpenVMS operating system to ensure transaction commits, aborts, and recoveries when failures occur. DECdtm services maintain database integrity through a two-phase commit transaction protocol (described in Section 7.2.3) to coordinate the flow of transaction requests and make sure that all of the operations of a transaction are performed or none of them are performed. If all operations are performed, the transaction is committed and the database is in a consistent state.

The Digital forms management package, DECforms (described in Section 5.4.1), integrates with VAX ACMS to provide forms processing capabilities in transaction processing environments. Data management is provided by DEC Rdb, DEC DBMS, and OpenVMS RMS.

7.2.3 Transaction Processing Support by DECdtm Services on OpenVMS Systems

DECdtm services available on every OpenVMS operating system ensure transaction and database integrity when used with multiple transaction processing monitors and database products in local and remote locations in distributed processing environments. With DECdtm services, application developers can define distributed transactions that include calls to Digital data management products.

DECdtm services comprise several components, including a transaction manager, interfaces to system services, logging and recovery services, and communication services.

The DECdtm transaction manager (TM) supports services, issued from the transaction processing applications, to start, end, or abort transactions. The TM coordinates the action of a distributed transaction by sending instructions to resource managers (such as RMS Journaling, DEC Rdb, or DEC DBMS) about how to complete the transaction. Each OpenVMS system ordinarily includes one DECdtm object, containing the TM for transactions initiated from that node.

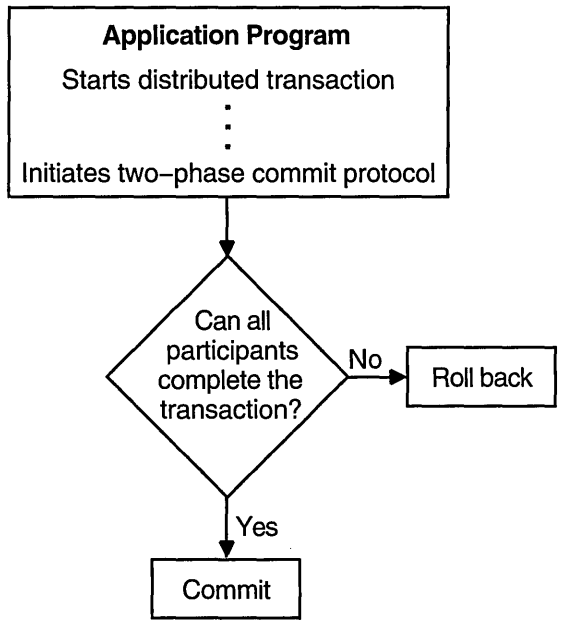
The DECdtm log manager provides a mechanism for storing a permanent record of the execution of distributed transactions. If there is a failure, a resource or transaction manager can read the log file to determine the state of the transaction at the time of the failure.

DECdtm implements a two-phase commit protocol to guarantee atomic transaction processing (see Figure 7–1). The two-phase commit protocol is a handshaking procedure in which all participants in a distributed transaction first agree to commit and then, on a signal from a coordinator, actually do commit. All resources commit or none of the resources commit; no operation is ever partially applied to a database. The commitment protocol permits the development of distributed TP applications that can survive the failure of individual CPUs and mass storage subsystems.

DECdtm provides the following data-integrity features:

- Ensures consistent execution of distributed transactions on the OpenVMS operating system.
- Ensures database integrity by guaranteeing that transactions spanning multiple RMS files and databases in local and remote locations complete as a single unit or not at all.

Figure 7–1 Two-Phase Commit Protocol for a Distributed Transaction


Application Program
Starts distributed transaction
Initiates two–phase commit protocol
Can all participants complete the transaction?
No
Roll back
Yes
Commit

ZK–1772A–GE

- Use of the two-phase commit protocol permits a large Rdb/VMS database to be divided into several smaller databases, and allows applications to access several different databases, without compromising the integrity or consistency of the data.
- Supported by multiple database products (DEC Rdb, DEC DBMS, DEC RALLY), transaction processing monitors (VAX ACMS), and RMS Journaling.

7.2.4 Other Distributed Transaction Processing Products

In addition to the DECtp system, the transaction processing capability includes the DEC RTR and, optionally, DSM.

DEC RTR is a distributed, fault-tolerant message routing system that supports several types of distributed applications, including TP. DEC RTR is designed to guarantee the delivery of a transaction message from a client to a server across a network for execution on distributed VAX systems anywhere in the world. Reliability is provided by software fault tolerance through redundant paths and redundant systems, if necessary. DEC RTR is suitable for customers with multiple geographic locations, large systems, the need to integrate distributed applications, and the requirement for the highest levels of reliability, availability, fault tolerance, and high-performance throughput. The global application integration and software fault-tolerant protection of DEC RTR ensures that server applications anywhere in the network are alway available to clients.

The optional DSM (Digital Standard MUMPS) system is available to build and support large, complex, distributed systems for many industries such as health care, banking, government, transportation, shipping, and manufacturing. DSM is a high-performance system, consisting of an interactive, interpretive programming language, database manager, and forms manager, that can be used to build, test, install, and maintain an application. VAX DSM is an open, callable system supporting interoperability with other layered systems.

7.3 Database Processing in Multivendor Environments

OpenVMS systems and optional products support the Digital information management strategy of making information on any system in the organization transparently available to any authorized user or application, regardless of which computers are used. The OpenVMS environment includes tools and software useful in developing and managing an open information network across an entire enterprise.

Information systems for general office computing, commercial applications, online transaction processing, or end-user applications are based on the database management system. OpenVMS supports a choice of database management systems for different uses: DEC Rdb, DEC DBMS, and RMS (RMS, which is integrated in the OpenVMS operating system, is described in Section 4.8).

7.3.1 DEC Rdb Database Management System

DEC Rdb is a relational database management system particularly appropriate for critical production systems in distributed, multivendor environments. DEC Rdb is designed to provide the high levels of data integrity, availability, and security required for commercial-strength applications. The high performance and high capacity provided by DEC Rdb are ideally suited to transaction processing and production applications.

DEC Rdb can store unstructured data, including laboratory, document, digitized voice, and graphic data as a single column in a table. This feature makes DEC Rdb appropriate for business, office, engineering, medical, and scientific applications.

The DEC Rdb relational database management system is based on SQL and can be used for distributed database applications. It is integrated with the OpenVMS operating system, making its physical location transparent to users and applications. It is also fully integrated with CDD/Repository, CDA Services, DECtp transaction processing, and COHESION software development products.

DEC Rdb supports open standards and provides full ANSI/ISO SQL compliance. The DEC Rdb SQL language has passed the National Institute of Standards Testing (NIST) tests to achieve full conformance with the FIPS SQL standard, including the SQL module language for C, COBOL, FORTRAN, and Pascal. In addition, DEC Rdb supports optimized SQL query and data-access capabilities. Conformance to standards has resulted in development of a large body of third-party solutions that can integrate with DEC Rdb.

DEC Rdb also supports the Multivendor Integration Architecture (MIA), developed by a consortium of vendors under the sponsorship of Nippon Telegraph and Telephone Corporation. DEC Rdb support for MIA includes the ability to specify character sets for database objects, the ability to specify more than one character set in a database, and support for multioctet character sets, including the Kanji character set as defined by the JIS X0208-1983 standard developed by the Japanese Industrial Standards Committee (JISC).

7.3.1.1 Production System Capabilities of DEC Rdb for OpenVMS

DEC Rdb for OpenVMS meets the needs of production system users as follows:

- High availability:
  - DEC Rdb provides for failover to another member of a VAXcluster if a member goes down.
  - Backup capabilities of DEC Rdb for OpenVMS permit backup while the database is being updated, overnight backup of very large databases, and incremental backup for transaction processing applications.
  - DEC Rdb can dynamically allocate disk space as files expand and can shrink files without taking the disk off line for restructuring.
- High performance capable of handling heavy online transaction workloads:
  - The robust database engine maximizes the potential of the OpenVMS environment (for example, DEC Rdb multiserver architecture permits use of symmetric multiprocessing and DEC Rdb on a VAXcluster can use the distributed lock manager to share files efficiently).
  - DEC Rdb can be tuned to avoid bottlenecks and use the most efficient data-access mechanism for a particular application.
  - Performance tools include DECtrace for performance monitoring and RdbExpert for designing efficient configurations.
- Data integrity and security:
  - DEC Rdb ensures the physical integrity of data, implementing locking, journaling, and transaction control to protect data against system failure, contention in update attempts, or program error.
  - DEC Rdb ensures atomic completion of transactions, performing a rollback if a transaction is interrupted by hardware or software failure.
  - DEC Rdb performs database journaling, using a recovery-unit journal and an after-image journal to permit rebuilding of a database after failure.
  - DEC Rdb, although not evaluated, is designed to meet C2-level security criteria as established by the National Computer Security Center (NCSC).
- Relational integrity: DEC Rdb supports ANSI/ISO integrity requirements, protecting applications and developers against possible mistakes such as deletion of data.
- Distributed database management: DEC Rdb utilizes DECdtm services to implement distributed transactions that ensure transaction and database integrity.
- Data connectivity: DEC Rdb can access data over the network using SQL/Services.

7.3.1.2 Database Interoperability Software

Digital provides integrated products that permit data sharing in heterogeneous environments. Many types of applications from multiple vendors on various platforms can exchange data and other functions across a network. Users can have a single, consistent interface to directly access major data sources in an organization.

DEC Rdbaccess solutions make corporate information on different kinds of database systems available to many users with no risk of corrupting or losing data. DEC Rdbaccess for RMS permits transparent read and write access to OpenVMS RMS files through the DEC Rdb SQL implementation. SQL/Services, a client/server component of DEC Rdb, provides a callable routine interface that lets users access DEC Rdb databases from other platforms (such as MS–DOS, Windows, Macintosh, OS/2, Sun, and ULTRIX systems) as well as from OpenVMS systems. Other products that provide Rdb SQL gateways are DEC Rdbaccess for ORACLE on OpenVMS (for read-only access to ORACLE databases on local or remote OpenVMS systems) and VIDA for DB2 (for read-only access to IBM DB2 databases).

DEC ACCESSWORKS software running on a preconfigured OpenVMS server (see Section 6.4.3.3) allows application programs running on various desktop computers to access information in many database systems over PATHWORKS network connections. DEC ACCESSWORKS supports the use of SQL to permit users to access data transparently from a variety of database management systems, including DEC Rdb, RMS, and ORACLE databases, and IBM DB2, VSAM, and IMS databases (see Figure 6–4). DEC ACCESSWORKS also provides tools for integrating other databases and software.

7.3.1.3 Database Tools Used in a Distributed Environment

A number of database tools support DEC Rdb database management:

- VAX Data Distributor: Manages the distribution of relational data among multiple nodes
- DECquery for MS–DOS and DECquery for Windows: Permits PC users to develop data queries without knowledge of SQL
- DECdecision: Uses the DECwindows interface to provide decision support applications
- DATATRIEVE: A structured, nonprocedural language that provides selective data retrieval, sorting, formatting, updating, and report generation for Rdb, DBMS, and RMS formats
- TEAMDATA: A fourth-generation language (4GL) decision support system with output in the form of tables, spreadsheets, and business graphics
- DEC RALLY: A fourth-generation language environment for generating Rdb applications, accepting data from multiple sources including DEC Rdb databases, TEAMDATA files, RMS files, and DATATRIEVE

Other data management tools used with DEC Rdb include:

- CDD/Repository services: Enable developers, users, and applications to create, store, and access common definitions for data, objects, and methods. CDD (Common Data Dictionary) data definitions can be obtained from multiple sources, including ORACLE and IBM VSAM and DB2 databases.

- Compound Document Services: Support the creation, display, processing, and interchange of different data types (including text, graphics, images, voice, tables, and multimedia) on different computing platforms. CDA (Compound Document Architecture) Services provide a consistent way for developers to build applications that can create, store, view, and share revisable compound documents among applications on multiple computing systems. (The CDA Viewer available with the DECwindows Motif user interface is described in Section 5.2.2.)

7.3.2 DEC DBMS Database Management System

DEC DBMS is a general-purpose, high-performance, CODASYL-compliant database management system designed to handle high transaction volumes. DEC DBMS works well for applications that require high throughput and the ability to handle complex relationships. It is particularly suitable for use when data relationships are complex but stable, when there are large amounts of data to be processed, when fast transaction processing is required, and when many concurrent users must be supported.

DEC DBMS is used to access and administer databases ranging in complexity from simple hierarchies to complex networks with multilevel relationships. It supports full concurrent access in a multiuser environment without compromising the integrity or security of the database.

DEC DBMS transaction capabilities include support for the two-phase commit protocol that operates in a distributed transaction environment in which the resource manager is DEC DBMS and the transaction manager is DECdtm. DEC DBMS also supports CDD/Repository, a distributed data dictionary that manages definitions shared by databases and applications across the network.

DEC DBMS also supports the Database Restructuring Utility (DRU), which provides the capability of changing many database characteristics without unloading and reloading the database.

A
OpenVMS Support for Standards

The OpenVMS operating system and related optional products support industry and international standards and specifications. This appendix provides summary information on OpenVMS conformance and compliance with standards and specifications adopted by recognized standards bodies, including the following:

- American National Standards Institute (ANSI)
  - Forms Interface Management System (FIMS) [proposed ANSI/ISO standard]
- Institute of Electrical and Electronic Engineers (IEEE)
  - Portable Operating Systems Environment Interface (POSIX)
- International Organization for Standardization (ISO)
- International Telegraph and Telephone Consultative Committee (CCITT) [recommendations]
- Japanese Industrial Standards Committee (JISC)
- National Institute of Standards and Technology (NIST)
  - U.S. Federal Information Processing Standards (FIPS)
- Open Software Foundation (OSF):
  - OSF Application Environment Services (AES)
  - OSF Distributed Computing Environment (DCE)
  - OSF Distributed Management Environment (DME)
- X/Open Portability Guide Issue 3 (XPG3)
- X Window System (Version 11)

A summary of the types of standards, draft standards, and specifications supported in major areas of OpenVMS capability appears in Table A–1.

Table A–1 OpenVMS Support for Industry and International Standards and Specifications

Technical Area	Standard
Operating System Information Interchange	ANSI, FIPS, ISO
Operating System Interfaces	IEEE POSIX 1003.1, P1003.4
	XPG3 BASE
User Interfaces	X Window System

(continued on next page)

Table A–1 (Cont.) OpenVMS Support for Industry and International Standards and Specifications

Technical Area	Standard
	OSF/Motif
	POSIX P1003.2
	XPG3 BASE
Small Systems Interconnect	ANSI SCSI
Local Area Networks (Ethernet and FDDI)	IEEE 802/ISO 8802 (CSMA/CD)
Management	OSF DME
	OSF DCE
	ISO (CMIP)
	ISO 9660
Database	ISO
	ANSI
	SQL
	CODASYL
	JIS
Data Interchange	ISO ODA
	ANSI X.12 (EDI)
	ISO SGML
Data Repository	ANSI IRDS
Languages	ANSI/MIL-STD./ISO Ada
	ANSI/ISO BASIC
	ANSI/ISO C
	ANSI/ISO COBOL
	ANSI/ISO FORTRAN, FIPS
	ANSI/ISO Pascal
Networking and Communication	ISO standards (OSI model)
	X.400 CCITT/ISO
	X.25 CCITT
	ISO FTAM
	TCP/IP
	NFS
	PostScript
	Display PostScript
Distributed Applications	OSF DCE
Distributed Transaction Processing	X/Open DTP XA
Graphics Interfaces	ISO GKS
	ISO GKS 3–D
	ISO PHIGS

(continued on next page)

Table A–1 (Cont.) OpenVMS Support for Industry and International Standards and Specifications

Technical Area	Standard
Forms Interfaces	ANSI/ISO FIMS (Proposed)

Index

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-DIGITAL (800-344-4825) and press 2 for technical assistance.

Electronic Orders

If you wish to place an order through your account at the Electronic Store, dial 800-234-1998, using a modem set to 2400- or 9600-baud. You must be using a VT terminal or terminal emulator set at 8 bits, no parity. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825) and ask for an Electronic Store specialist.

Telephone and Direct Mail Orders

From	Call	Write
U.S.A.	DECdirect Phone: 800-DIGITAL (800-344-4825) FAX: (603) 884-5597	Digital Equipment Corporation P.O. Box CS2008 Nashua, NH 03061
Puerto Rico	Phone: (809) 781-0505 FAX: (809) 749-8377	Digital Equipment Caribbean, Inc. 3 Digital Plaza, 1st Street Suite 200 Metro Office Park San Juan, Puerto Rico 00920
Canada	Phone: 800-267-6215 FAX: (613) 592-1946	Digital Equipment of Canada Ltd. 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6 Attn: DECdirect Sales
International	————	Local Digital subsidiary or approved distributor
Internal Orders[1] (for software documentation)	DTN: 241-3023 (508) 874-3023	Software Supply Business (SSB) Digital Equipment Corporation 1 Digital Drive Westminster, MA 01473
Internal Orders (for hardware documentation)	DTN: 234-4325 (508) 351-4325 FAX: (508) 351-4467	Publishing & Circulation Services Digital Equipment Corporation NR02-2 444 Whitney Street Northboro, MA 01532

[1]Call to request an Internal Software Order Form (EN–01740–07).

Reader's Comments

OpenVMS Software Overview
AA–PVXHA–TE

Your comments and suggestions help us improve the quality of our publications.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (product works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

I would like to see more/less ______________________________

What I like best about this manual is ______________________________

What I like least about this manual is ______________________________

I found the following errors in this manual:

Page Description

Additional comments or suggestions to improve this manual:

For software manuals, please indicate which version of the software you are using: ______________

Name/Title ______________________ Dept. __________

Company ______________________ Date __________

Mailing Address ______________________

______________________ Phone __________

---- **Do Not Tear – Fold Here and Tape** --

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OpenVMS Documentation
110 SPIT BROOK ROAD ZKO3–4/U08
NASHUA, NH 03062–2642

---- **Do Not Tear – Fold Here** --

Reader's Comments

OpenVMS Software Overview
AA–PVXHA–TE

Your comments and suggestions help us improve the quality of our publications.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (product works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

I would like to see more/less __________

What I like best about this manual is __________

What I like least about this manual is __________

I found the following errors in this manual:

Page Description

Additional comments or suggestions to improve this manual:

For software manuals, please indicate which version of the software you are using: __________

Name/Title __________ Dept. __________

Company __________ Date __________

Mailing Address __________

__________ Phone __________

Do Not Tear - Fold Here and Tape

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OpenVMS Documentation
110 SPIT BROOK ROAD ZKO3–4/U08

NASHUA, NH 03062–2642

Do Not Tear - Fold Here