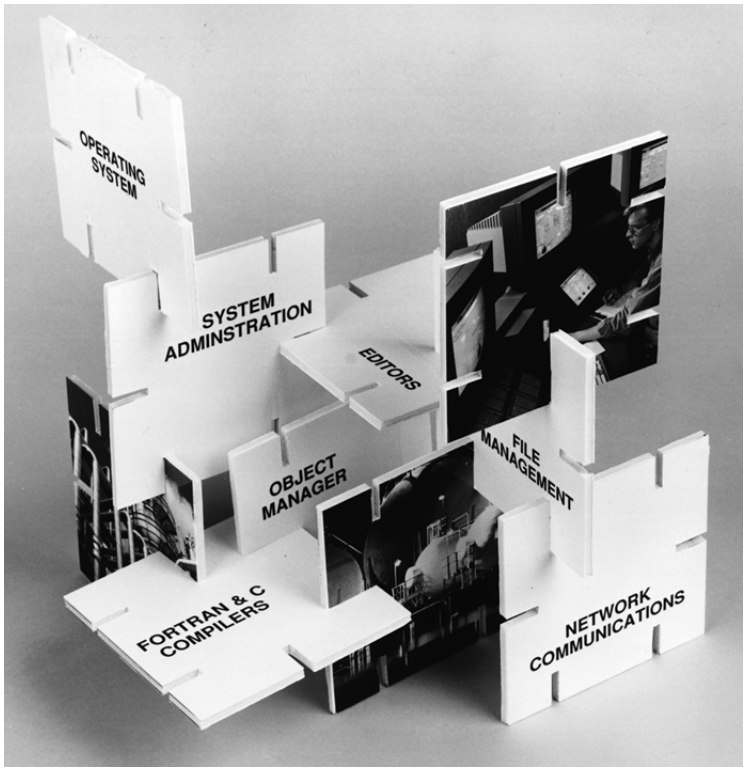
FOXBORO®

PSS 21S-2B2 B3

I/A Series® Software
50 Series Operating Software



The operating software is a set of programs that control and organize the activity of the I/A Series system. These programs direct the activities of the system modules, manage multiusers and multitasking, and manage the system's files without user participation or supervision. The operating software includes the operating systems and other subsystems, such as Inter-Process Communications, the Object Manager, and other Application Programming Interfaces (API).

The predominant trends in today's computing industry are toward openness, the ability to link industry-standard hardware and software from many different manufacturers, distributed client-server computing, the ability to transparently harness the power of all processors on a network while running applications from any given station location.

The SunSoft (Solaris 2.X) operating system provides workstation and application processor solutions based on an open computing philosophy. 50 Series stations allow information to flow seamlessly among computer systems from different manufacturers. The benefits are many – protection of current hardware, software, development, and training investments; a broader choice of high-quality software; and the ability to use the best computing solution for a task. With 50 Series stations, users are free to choose industry-standard solutions that best suit their needs without being tied to proprietary-hardware, operating-system, or networking architectures.

The introduction of higher performance 50 Series workstations and application processors is transparent to Foxboro, user, and third-party application software.

OPERATING SYSTEM

The operating system for 50 Series stations (AP51/AW51/WP51) is the SunSoft Inc., Solaris 2.X. Solaris 2.X is based on UNIX System V Release 4 with a number of significant SunSoft improvements. It is a multitasking, operating system with extensive functionality in areas such as symmetrical multiprocessing, real-time extensions, increased security, enhanced network performance, and improved system administration. To help both users and developers anticipate future technologies and preserve investments, the operating system offers worldwide industry standards. These standards include the Institute of Electrical and Electronics Engineers (IEEE) POSIX 1003.1 and IEEE 754 ANSI C, X/Open Portability Guide (XPG3), AT&T System V Interface Definition (SVID) Issue 3, SPARC Compliance Definition (SCD) 2.0, ISC 9660. Solaris also conforms to the following standards: SVR4 Generic ABI, SVR4 SPARC ABI, SVR4 DDI/DKI.

A symmetrically multiprocessing (SMP) and multithreaded (MT) operating system kernel provides increased performance and optimizes overall throughput and application response time. The kernel directs processor activities on a prioritized, pre-emptive basis. Pre-emptive means that higher-priority interrupts are handled immediately. This gives the processor multitasking operation. Since the kernel is memory resident and uses pre-emptive scheduling, it can perform multitasking operations at a very fast response rate.

The SunSoft real-time enhancements to the real-time extensions of UNIX SVR4 include such features as fixed priority real-time processes, user process priority manipulation, high resolution timers, completely pre-emptive scheduling, process priority inheritance, and deterministic and guaranteed dispatch latency.

Enhanced security options include an automated security enhancement tool (ASET), a shadow password file, and authentication modes. With ASET, administrators can increase a system’s security level, provide warnings for security hazards, and perform system file integrity checks. Shadow passwords allow for enhanced password aging and login controls as well as storage of encrypted passwords in a separate unreadable file.

In addition, administrators can choose which of three authentication modes is appropriate for their site: UNIX authentication, secure RPC, or Kerberos.

Solaris 2.X’s core networking technology, ONC+, is a widely used heterogeneous networking solution providing an extensive family of protocols and distributed services.

The Distributed System Architecture (DSA) allows access to resources (files, message queues, applications, databases, printers, and so forth) that are distributed throughout the network. Since this system is merged with the network, a number of communications problems are simplified:

- The operating system does not need to transfer an entire file to a local processor just to read part of it.
- Only one copy of a file need exist on a network.
- The resources available to every processor are the sum of the resources on all processors.
- Tasks on different processors can synchronize and exchange messages.
- Since utilities do not need to be changed to use the network, the network is transparent to the utilities and to programs that use them.

File and Memory-Management

The operating system includes an integrated file and memory-management facility that efficiently manages system resources and system interfaces, enabling programs to gain access to files through address mapping techniques.

The operating system constructs a system's virtual memory out of the storage resources available to that system, including file systems on locally attached storage devices as well as those available through network access protocols. The system uses techniques such as demand paging and process swapping to adapt to increasing workload demands on primary memory. Demand paging allows the system to use disk space as pages of main memory.

The core of the virtual memory system is a suite of routines that map files (including devices, ports, and so forth) into the virtual address space of a process. This allows a process to access a file as locations in memory instead of having to use file operation and system calls. More than one process can map the same file, and with shared memory – another feature of virtual memory – the same copy of the file is mapped into each process. In this way, a task running on any station can make file access calls both locally and remotely (over the network). The file system is hierarchical; that is, it consists of directories that can contain files or subdirectories. Each physical input/output (I/O) device, typically a disk or tape drive, to or from main memory, is treated as a file, allowing consistent file and device I/O.

The file management system allows synchronous access to the files. Synchronous operation implies that the task that makes the request suspends while the operating system performs the access. The task does not resume until the access completes (successfully or in error).

The file structure is based on the Berkeley fast file system, which uses 24-bit logical disk-block pointers and 8192-byte disk blocks. A single file can grow dynamically up to a maximum of two gigabytes. The file access calls are based on the standard UNIX file access calls.

File management uses a distributed file system; that is, stations with bulk storage devices act as file servers for other stations. These file servers are called Application Processors (for example, AP51) or Application Workstations (for example, AW51). APs and AWs are self-hosting and contain the entire shell and kernel portions of the operating system shown in the simplified Figure 1.

For enhanced system file security, the Application Processor and the Application Workstation can optionally use mirrored hard disks. Disk mirroring is based on the powerful Metadisk technology – a group of components accessed as a single device. This allows on-line disk replacement/recovery should one of the mirrored disks fail.

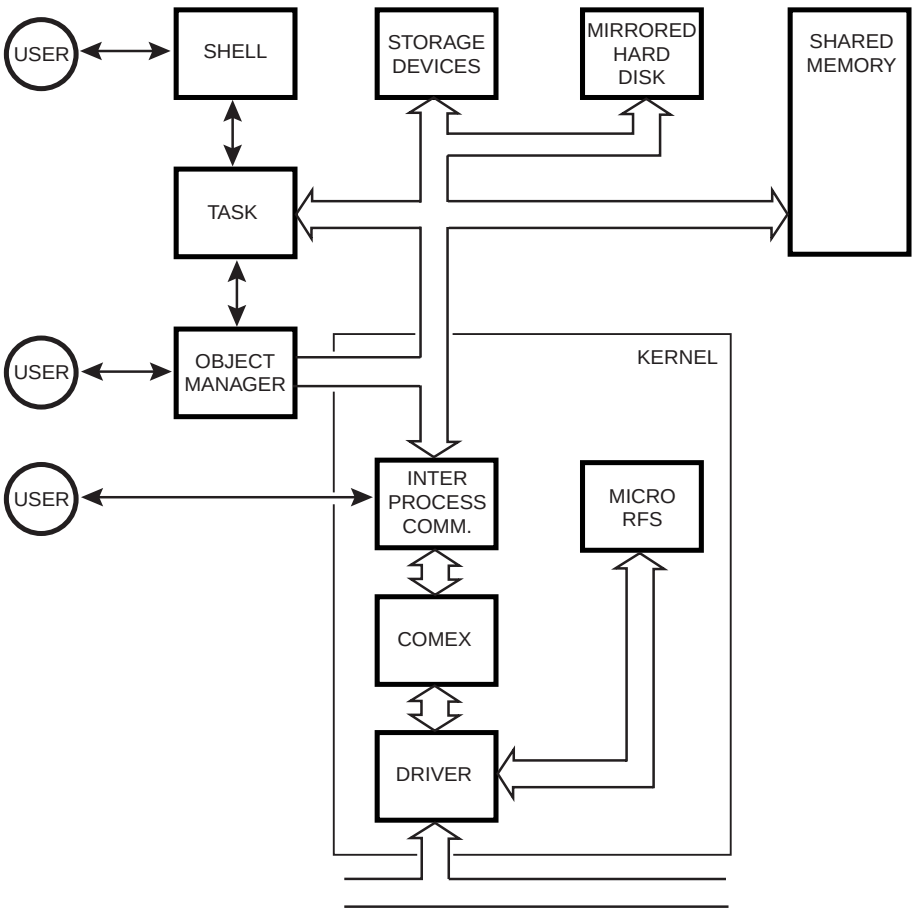


Figure 1. Simplified 50 Series UNIX Operating System Block Diagram

Workstations (that is, WP51, AW51) contain their own file system for local displays. A file management utility facilitates easy distribution of display file updates between multiple APs and WPs by directory, file name, source, and destination file date. Workstation Processors are self-hosting and contain the same kernel portion of the operating system as Application Processors, except for disk mirroring, but also include user interface – keyboard and annunciator software.

Dynamic Linking and Shared Libraries

With dynamic linking, libraries are loaded separately from the application and only one copy is required in memory at one time. Programs are dynamically linked with system services and library routines, providing ease-of-update to application piece parts. Libraries are provided in dynamically linked forms including libraries that interface the system to applications. Window systems are also provided in shared library form, to facilitate sharing of code common to different applications. Compared to the traditional archive form of system libraries, the operating system makes more efficient use of system memory and disk resources, leaving more memory available to applications.

Loadable Modules

Loadable modules are programs that can be executed as though they had been built with the kernel. They are device drivers and other kernel routines that the kernel can use without having been statically linked into the kernel.

When a module is first referenced, it is added to a running system. For example, not until data is sent to a set of ports is the driver that controls access to the ports loaded. This allows the kernel to be reconfigured dynamically. Developers can change one kernel module without having to change any other, and modules can be added and removed without having to rebuild the entire kernel. This makes custom driver development and installation easier.

Networking Capabilities

The operating system's Distributed System Architecture (DSA) supports networking. Built into the system are the capabilities of keeping track of which processors are attached to which node, and how to (and who can) access them. The Machine Access Table and the User Access Table determine the configuration of the network.

There are two network file system types: Network File System (NFS) and Micro Remote File System (µRFS). NFS provides the ability to share files across a network in a heterogeneous environment. µRFS supports remote file sharing and works on the directory level. When a client accesses data in µRFS mounted directories, the data is cached in local memory where it can be accessed by local processes. This substantially reduces network traffic.

External Data Representation (XDR) specifications provide a common way of representing a set of data types over a network allowing different structure alignment algorithms to share a common format over the network. This provides compatibility over the network for all present and future I/A Series stations. This specification also enhances data format translation with optionally connected systems over an information network.

User Access

User access permission to remote processors is handled by user and group ID mapping tables. Before a task can be run on a remote processor, a task checks the ID mapping tables to make sure that the user and group ID's are allowed access before a connection is made. If no ID mapping table exists for the connecting processor, access is denied to all users of that processor.

Error Handling Subsystem

The error handling subsystem allows the system to recover from and report an error that occurs in the system. It does this either by taking the necessary action itself, or by passing information to other subsystems so they can handle the error.

Errors handled by the System Management subsystem are failures in hardware detected by a combination of error detection hardware and diagnostic software.

Errors handled by the operating system are software errors that can be due to design errors in user-written software, unavailable resources, and other run-time contention errors. The operating system works in conjunction with the System Management subsystem to report these errors.

Error Detection and Reporting

Software run-time errors are detected and handled in several ways. Calls to the operating system are checked to see that all the call parameters are within limits. Whenever the operating system encounters an error, it sends an interrupt to the calling task.

When a task receives the interrupt signal, it can invoke a system subroutine to take the appropriate action (for some classes of errors) or terminate.

The error-reporting function can be invoked to report errors in a consistent format to the specified output device.

Error Analysis

Each station can save the relevant text and data areas and the contents of the processor's registers when a run-time error occurs. You can also trigger this "dump" by code within the task (for example, for debugging). The dump is automatically written to a file and a message to that effect is printed.

There is also software for Foxboro to analyze the dump file for any processor in the system. This software can interpret the state of the various data structures within the operating system, do run-time stack backtracking, and print a listing of various sections of memory for interpretation.

System Priority Scheduling

The process is the basic unit that is scheduled. A process is a program image and its execution environment (register values, status of open files, current directory, and so forth).

The scheduling system schedules processes according to their priority. Priority for processes is a value calculated using the process state, processor use time, and "nice" value.

The priority of processes is evaluated every second and the priority can change. When the priority is recalculated, the calculation lowers the priority (increases the priority number) for processor use time and higher nice values. It raises the priority (decreases the priority number) for time spent waiting to run.

Processes in the sleeping state assume the priority of the event they are waiting for, which is always a higher priority than running processes. Giving a higher priority to processes that use a lot of input/output resources helps keep resources available.

The scheduler always switches to the process with the highest priority. Ties go to the process that has been waiting the longest (first in, first out).

Utilities, Libraries, and the Command Language

The operating system provides three command shells: the Bourne shell, the C shell, and the Korn shell. These are command interpreters with high-level programming-language constructs (IF-THEN, DO-WHILE, and so forth). You can create command files to perform tedious or repetitive command sequences.

The operating system has commands associated with or supporting each of the following:

- Text editing and formatting
- Data sorting
- System administration
- UNIX file management
- Distributed System Architecture (DSA)
- Virtual Terminal Emulation

Support for Application Programming Interfaces (API):

- Inter-Process Communications
- Remote Procedure Call (rpc)
- Object Manager
- Display Manager/Hi Library
- Historian
- Real-Time Relational Database Manager (ISQL, ESQL/C)
- X-Window (X11)
- Open Look Window Manager
- Application Interface Software (AIS)

WINDOWING SYSTEM

The 50 Series workstations include support for the intuitive, icon-based graphical user interface, delivering a common look and feel across applications and hardware platforms. An industry-standard window system, X11, and a proven development toolkit, provide advanced windowing functionality. The Open Look Window Manager provides a common intuitive look and feel on top of the existing "pick and point" operation of the I/A Series system. It also provides window management for other integrated third party applications. If optional network communications is supplied with the workstation, X-Window displays from other connected systems can be integrated among other I/A Series display windows.

TIMING FUNCTIONS

The timing functions include a system clock interrupt handler, a station clock for time and date, and a user interface for scheduling tasks.

Each station provides user tasks with facilities for dealing with time of day and for software timers. These timing facilities are driven by interrupts coming in from a real-time clock. The operating system uses these real-time clock interrupts for time-of-day scheduling functions, timeout functions, and notification functions such as wake-up signals.

System Clock

The system clock has a resolution of 10 milliseconds. Values of the system clock can be read in one microsecond intervals. The network time-of-day enables all stations to synchronize their clocks to within one-tenth of a second. Any user task can read the time of day, and the System Management subsystem provides a facility to change it.

Station Clock

The Application Processor stations are configured to act as master timekeepers. Other station processor's real-time clock chips maintain station times between updates from the master timekeeper.

Clock Timers

The system provides software timers that user tasks can use to:

- Acquire a timer for its use
- Specify the time-out period of the timer
- Enable the timer
- Disable the timer
- Deallocate the timer.

Time-of-day Scheduling

You can schedule or deschedule a task and list currently scheduled tasks. Tasks can be scheduled to:

- Run once at a specified time of day.
- Run periodically at a specified interval (regardless of changes in station time).
- Run periodically at a specific time of day.

OPEN SYSTEMS INTERCMONNECTION

The system Transport Layer Interface (TLI) provides support for the model of Open Systems Interconnection (OSI).

Layer 7	application
Layer 6	presentation
Layer 5	session
Layer 4	transport
Layer 3	network
Layer 2	data link
Layer 1	physical

A basic principle of the reference model is that each layer provides services needed by the next higher layer in a way that frees the upper layer from concern about how these services are provided.

The transport layer in the reference model provides the basic service of reliable, end-to-end data transfer needed by applications and higher layer protocols. In doing so, this layer hides the topology and characteristics of the underlying network from its users. More important, however, the transport layer defines a set of services common to layers of many contemporary protocol suites, including the International Standards Organization (ISO) protocols, the Transmission Control Protocol and Internet Protocol (TCP/IP) of the ARPANET, Xerox Network Systems (XNS), and the Systems Network Architecture (SNA).

An inherent characteristic of the transport layer is that it hides the details of the physical medium being used. The Transport Interface offers both protocol and medium independence to networking applications and higher layer protocols. The UNIX system Transport Layer Interface is modeled after the industry standard ISO Transport Service Definition (ISO 8072).

OBJECT MANAGER

The Object Manager is a subsystem that controls access to data units called objects. There are two classes of objects:

- Control and I/O Objects (compound:block.parameters)
- Shared Objects
  - Processes (Tasks)
  - Devices
  - Aliases (character strings)
  - Variables (integer, character, long integer, floating point, and string)

The Object Manager acts as an interface between applications and the data they require. It keeps track of the locations of objects so you can write applications that access data by name only, without knowing the system configuration or network locations.

Because of the Object Manager, object databases can be created, modified, or moved to another station without having to modify any of the applications that access the data. You can write an application before creating the database it uses. The Object Manager allows applications to:

- Create, locate, and delete objects
- Get or set single object values (such as process variables)
- Read or write sets of shared objects (variables)
- Receive notification when an object value changes by a specified amount.

INTER-PROCESS COMMUNICATIONS

"Process," in this usage, refers to the UNIX term for tasks and their executing environment, rather than the manufacturing or production process. In addition to the standard Inter-Process Communications, The Foxboro Company has added extensions which provide for distributed communications between stations in the system.

These extensions allow for messages that can:

- Pass data or control information between tasks
- Notify another task of an event
- Synchronize the use of common network resources
- Communicate with applications distributed throughout the system, without necessarily knowing where they are.

Inter-Process Communications supports not only connection-oriented communications, but also connectionless and broadcast communications. The user interface to Inter-Process Communications is through a series of C language subroutines.

LANGUAGE PROCESSING

The operating system supports the C and FORTRAN programming languages, including compilers, linkers, debuggers, and semantic checkers. All system libraries are fully supported for C language. Object manager libraries are supported for FORTRAN and C language.

The operating system supports high-performance compilers for C and FORTRAN languages. The languages share a common language base, composed of optimizers, code generator, and libraries. All languages have access to floating-point processing for high-precision computations which adhere to the ANSI/IEEE floating-point standard. Both FORTRAN and C language compilers are optional.

OPERATING SYSTEM FEATURES SUMMARY

Operating System Standards
- Institute of Electrical and Electronics Engineers (IEEE) POSIX 1003.1
- AT&T System V Interface Definition (SVID) Issue 3
- SPARC Compliance Definition (SCD) 2.0
- Transmission Control Protocol and Internet Protocol (TCP/IP) for the ARPANET, Xerox Network Systems (XNS), and the Systems Network Architecture (SNA)

Features shared by Languages
- Global optimization
- Peephole optimization
- IEEE-Standard Floating Point (ANSI/IEEE 754) including support for accelerators
- Common code generation
- Interlanguage calling
- Shared libraries
- Access to UNIX operating system calls
- Common set of profiling tools
- Variety of other programming support tools
- Access to graphics environment
- Access to data communications

Window Features
- X/Open Portability Guide XPG3
- X11 window system
- Open Look Window Manager

File System Security
- Optional Disk mirroring
- Metadisk technology

FORTRAN (Optional) Features
- ANSI X3.9-1978 FORTRAN
- ISO 1539-1980
- MIL-STD 1753
- FIPS 69-1 BS6832
- IEEE 754 Floating Point Arithmetic Standard
- VAX VMS FORTRAN 5.0 extensions
- Cray Supercomputer extensions
- Four optimization levels
- Interlanguage calling with C compiler
- Floating license management

C (Optional) Features
- Key ANSI language features
- ANSI C X3.159-1989
- System V Interface Definition (SVID)
- X/Open (XCS-QUE-3.106)
- IEEE 754 Floating Point Arithmetic Standard
- Switch selectable ANSI C / K&R C/ mixed mode compatibility
- Four optimization levels
- Interlanguage calling with FORTRAN compiler
- Floating license management

Standard, Advanced APIs
- Inter-Process Communications
- Remote Procedure Call (rpc)
- Object Manager
- Display Manager/Hi Library
- Historian
- Real-Time Relational Database Manager (ISQL, ESQL/C)
- X-Window (X11)
- Open Look Window Manager
- Application Interface Software (AIS)

The Foxboro Company
33 Commercial Street
Foxboro, Massachusetts 02035-2099
United States of America
http://www.foxboro.com
Inside U.S.: 1-508-543-8750 or 1-888-FOXBORO (1-888-369-2676)
Outside U.S.: Contact your local Foxboro Representative.

MB 021 Printed in U.S.A. 0598