Sun Technical Bulletin
March 1993

Technical Information Services

Sun

Sun Microsystems Computer Corporation
A Sun Microsystems, Inc. Business

2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

Part No: 812-9303-01
Issue 1993 - 03

The *Sun Technical Bulletin* is distributed to customers with software/hardware or software-only support contracts. Send comments or corrections by electronic mail to *stb-editor@Sun.COM*, or by mail to:

Sun Technical Bulletin
Sun Microsystems Computer Corporation
2550 Garcia Avenue , M/S MPK02-04
Mountain View, CA 94043-1100 U.S.A.

Contents

1. STB Editor's Introduction 1-1

STB Overview 1-1

Updates 1-4

ISDN Technology (October 1992) 1-4

Performance Improvements (September 1992) 1-6

2. Solaris 2-1

Introduction to the ToolTalk Service 2-1

Introduction 2-1

Universal Plug-and-Play:
The Application Integration Requirement 2-2

The ToolTalk Service. 2-4

Benefits of the ToolTalk Service 2-5

Using the ToolTalk Service 2-6

ToolTalk Messages 2-9

Standard Message Sets. 2-12

Moving Toward Distributed Objects 2-15

3. SunOS **3-1**

Swap Meet 3-1

What's in a Name? 3-2

Leave Room for Later 3-3

Use What You Take 3-3

Dealing With No-Shows 3-5

4. Security **4-1**

Solaris Basic Security Module (BSM) 4-1

Introduction 4-1

Administering Auditing 4-1

Device Allocation 4-19

5. System Administration **5-1**

New Administration Tools with Solaris 2.1 5-1

Changes to the Administration Tool User Interface 5-2

Printer Manager 5-3

User Account Manager 5-11

6. Hardware **6-1**

Sun Type 5 Keyboard 6-1

Power and Audio/Monitor Keys 6-3

Keycode Assignments 6-4

R-Key Support 6-6

Comparative Mappings for the Right Keypad 6-7

Line Feed Key 6-8

L-Key Labels 6-8

Diagnostic Codes 6-9

Questions and Answers 6-10

7. Windows **7-1**
SunSoft's Windows Directions 7-1
OLIT Improvements. 7-1
Why a New Direction?. 7-2
Questions and Answers. 7-2
8. Software Release Versions. **8-1**
Current Sun Software Products and Release Versions 8-1
Operating Systems 8-1
Window Systems & Desktop Utilities 8-1
Graphics Software 8-2
Software Development 8-2
System Management 8-3
Other Products 8-3
Connectivity Products 8-4
Updated Sun Operating System Support Matrix 8-5
9. Notes & Comments **9-1**
World Hotlines 9-1
Reporting Bugs 9-3
Submitting Software Bugs: United States and Canada . . 9-3
Submitting Software Bugs: Europe 9-4
Submitting Software Bugs: Intercon. 9-8
Your STB Subscription 9-10
Index

STB Editor's Introduction 1

STB Overview

This month's STB contains articles of general interest in the topics outlined below.

Please note the *Updates* section at the end of the chapter. This section features new or updated information about articles from past issues of the STB. We appreciate the participation of our readership in calling this information to our attention.

Solaris

"Introduction to the ToolTalk Service" is a SunSoft technical white paper that describes ToolTalk's universal plug-and-lay capability, enabling independently developed applications to communicate and cooperate. The ToolTalk service is a standard feature of the Solaris 1.*x* and 2.*x* releases, so all Solaris users have ToolTalk installed and available.

SunOS

"Swap Meet" is another in our series of reprints of Hal Stern's columns from *SunWorld* magazine. This article discusses changes in requirments for swap space in Solaris 2.*x*, and how to plan for both average swap space usage and peak demands.

Security

"Solaris Basic Security Module (BSM)" is excerpted from the manual that accompanies the newly-released BSM product. BSM supplies the security features required by the C2 class of the Trusted Computer System Evaluation Criteria (TCSEC) that are not included in the standard Solaris 2.1 release: a security auditing subsystem, and a device allocation mechanism that provides object reuse characteristics for removable or assignable devices. Other required C2 features are provided by the standard Solaris 2.1 system.

System Administration

This month's article, "New Administration Tools with Solaris 2.1", is excerpted from the SunSoft Press book *Solaris System Administrator's Guide* by Janice Winsor of SunSoft, which was published in January and is available in retail bookstores worldwide. The article describes changes to the Administration Tool (`admintool`) and introduces two new applications that are available with the Solaris 2.1 Administration Tool: Printer Manager and User Account Manager.

Hardware

"Sun Type 5 Keyboard" describes the new Type 5 keyboard design that currently ships with SPARCstations, and illustrates its differences from the previous generation of Sun keyboard. The Type 5 keyboard is available in an alternative UNIX version which retains certain key locations from the older Type 4 keyboard.

Windows

"SunSoft's Windows Directions" is an announcement from SunSoft regarding future OpenWindows policies in light of Sun's recent agreement with Adobe Systems to license the Display PostScript System (DPS) for Solaris, and in light of Sun's commitment to MIT's X Window System. SunSoft will soon ship an OpenWindows product based on the MIT source code.

The last release of Solaris featuring X11/NeWS and TNT (The NeWS Toolkit) will ship in mid-1993. Subsequent releases of Solaris will incorporate MIT's X11R5 coupled with Adobe Display PostScript Level 2. Applications using NeWS imaging must be ported to DPS, and SunSoft is recommending that applications using TNT be ported to the OPEN LOOK Intrinsic Toolkit (OLIT).

Software Release Versions

This section includes the current Sun software products and release version tables. We also include a revised updating of Sun's support matrix for Operating System release levels and platforms.

Notes & Comments

World Hotlines

For Sun customers world-wide served by your local service groups, use the customer service telephone numbers listed in this monthly item. Also, look to this section during the upcoming year for details on your local support call policies and procedures.

Reporting Bugs

The information in this monthly section continues to be expanded as Sun software service centers are added world-wide. The section includes a list of Sun service centers, addresses, e-mail hotlines, telephone hotlines, and FAX numbers.

Your STB Subscription

This notice is published monthly, giving customers useful information regarding ordering multiple STB hardcopy subscriptions, and duplicating additional STB copies from a single "master" copy. This duplication permission is limited, as detailed in the note.

Updates

This section presents updates on subjects covered in past issues of the Sun Technical Bulletin.

ISDN Technology (October 1992)

A reader comments:

> I was not able to figure out from the contents of the article if the ISDN interface is based on DLPI, or in other words, if there is an ISDN DLPI STREAMS device driver. It's also not clear to me how the ISDN interface is integrated in the overall UNIX system.

Replies from the field:

> DLPI (Data Link Provider Interface) is a standardized kernel interface usually used by the driver modules.
>
> Within SunLink ISDN 1.0, there are several STREAMS modules in the kernel. But there are also other modules which are not in the kernel. The physical driver is in the kernel but does not conform to DLPI. The D-channel software is implemented as a user process and therefore does not use DLPI. So in one sense, the ISDN interface is not based on DLPI.
>
> On the B-channel, there are some modules in the kernel which implement DLPI, but they are not really accessible to the end user.
>
> For SunLink ISDN 1.0, our first ISDN product, it's difficult to talk about an "ISDN interface": the goal is to provide transparent access to ISDN using the TCP/IP protocol family. Really, the end user does not directly interface with ISDN. He uses TCP/IP, `rlogin`, `telnet`, X11, etc., which will go through ISDN if necessary.
>
> One might expect that an ISDN device driver would provide a DLPI interface. However, the DBRI device driver only supplies a physical layer interface. It does not supply a "Data Link" interface — so reference to the "Data Link Provider Interface" is not entirely appropriate. The DBRI driver does support the `M_DATA` message of the DLPI which is also part of conforming to the `streamio` interface.

If one were to write a LAPD STREAMS multiplexor and link one of the DBRI's D-channel devices under LAPD, then it would be appropriate for the LAPD mux to present a DLPI interface to its users. Call control is built on top of LAPD/D-channel, and other data link interfaces can then be built using call control and the B-channels provided by the device driver.

In the SunLink ISDN product, there are two user-level processes that cooperate to provide IP connectivity over ISDN Basic Rate B-channels. One process handles the LAPD and call control responsibilities; the other process connects between call control, B-channels, PPP, and IP. The second process is called the ISDN Manager.

For the time being, SMCC's SunLink ISDN gives access to ISDN connections using the IP interface. Direct access to the ISDN driver is also possible — see the man pages for `isdnio`.

In the future, we will provide direct ISDN interfaces to give direct programmatic access over ISDN. Access to the ISDN device will be available via the Solaris Teleservices API, a high-level interface for accessing various connection and conferencing technologies. This should be available in Spring or Summer of 1993, and will probably be bundled in the October '93 Solaris release.

Note – Within the next few months, there should be another white paper describing the Solaris Teleservices architecture and how ISDN fits in it. We will try our best to reprint this paper in the STB.

Performance Improvements (September 1992)

A reader comments:

> In the September issue of the STB, you have an article on performance in which you talk about some `vmstat` output (page 4-3). The statement is made that "anything over about 100 per second generally is considered to be a high context-switch rate ... in the example above, the context-switch rate is high (`116/68`) because the system-call rate is astronomical (`673/721`)." This doesn't seem very meaningful without taking into account the processor speed, number of context registers, etc.

Hal Stern replies:

> The comment raises a good point, that the actual *cost* of context switches and system calls is a function of processor speed and CPU design. Obviously, a CPU built to handle a large number of contexts has a lower context-switch cost than one, say, that only deals with one context at a time. The various Sun CPUs and MMUs have had support for anywhere from 8 to 65k contexts, with widely varying context switch times as well.
>
> But the "reasonable maximum" figure aims at a level above this. Instead of determining the highest possible rate, I was trying to give a feel for a "danger sign" — and 100 context switches per second means that no process is doing an involuntary context switch as a result of running out its time slice (might be interrupted). All of your processes are jumping on the CPU, doing some small amount of work, and jumping off to either give up the CPU or to be interrupted by another process that just finished blocking on I/O. It's this "ping pong" case that is a poor performer. It doesn't matter how fast the context switches are — whatever is causing them is consuming a large amount of system resources,and can usually be made more efficient.
>
> Sometimes a large context switch rate is nothing to worry about. If you're running a DBMS and have tons of backend processes doing asynchronous I/O for you, then you want lots of context switching, because this indicates a good transaction run rate through the system.

We would like to encourage our readership to e-mail us any comments or suggestions regarding the content of the Sun Technical Bulletin, including requests for articles on specific subject matters. We are aiming at a broad spectrum of topics that are of interest to all sectors of our audience, both highly experienced technical people and relatively inexperienced users who are new to desktop workstations. We are not the appropriate channel for researching any and all technical problems that may come up in the field, but we will do our best to follow up on questions about articles that we have printed in the recent past.

Please let us know what you think, and we will do our best to accommodate you in future issues.

Thanks.

The STB Editor
sun!stb-editor or *stb-editor@Sun.COM*

Solaris 2

Introduction to the ToolTalk Service

This article is a technical white paper from SunSoft, Inc.

Introduction

As computer users increasingly demand that independently developed applications work together, “interoperability” and “integration” are becoming important themes for software developers. These terms convey the idea of inter-application cooperation — of applications achieving more by working together than by working separately. SunSoft’s ToolTalk service, a network-spanning inter-application message system, has been designed to promote cooperation among independently developed applications.

Use the ToolTalk service when you want *universal plug-and-play* capability. The term *universal plug-and-play* means that any tool can be replaced by any other tool that follows the same protocol. That is, any tool that follows a given ToolTalk protocol can be placed (plugged) into your computing environment and can perform (play) those functions indicated by the protocol. Tools can be mixed and matched, without modification and without having any specific built-in knowledge of one another. Universal plug-and-play is built on the concept of describing applications by the functions the addressee needs to perform, rather than calling a specific type of application.

The ToolTalk service is completely distributed and network-aware. Location issues are entirely transparent to a user. While it provides easy global access, ToolTalk hides all networking details. The ToolTalk service can function as a partner to OpenWindows or can operate independently of any windowing system. In addition to this, ToolTalk messages can be retrofitted to any existing application.

The ToolTalk service is a standard feature of the Solaris 1.*x* and 2.*x* operating systems software platforms. All Solaris users have ToolTalk installed and available on their systems.

ToolTalk messaging is an API of the DOE (Distributed Objects Everywhere) Application Framework. If you adopt ToolTalk message compliant applications today, you can be assured of having made an excellent investment. You will be able to continue to use ToolTalk messaging in your existing applications with the DOE Application Framework as you enhance your applications to take advantage of DOE distributed object facilities.

Universal Plug-and-Play: The Application Integration Requirement

Today, computer users have access to a wide variety of applications that are either purchased from software suppliers or developed in-house. Not only do users want to use applications they are already familiar with, they want their tools of choice to be integrated with other applications automatically. Typically, a company limits the choices available to its workers by standardizing on one or more applications of a particular type. Unfortunately, this results in little integration among applications. This is not the ideal solution. Without the best, highly integrated tools, an organization does not have an edge in the global market.

Users want any and all applications to integrate automatically and consistently. They need automated access to all data types including numerical, text, graphical, audio, video, and control information. And they should not be required to know where the information is stored. This is the *universal plug-and-play* requirement of applications, a theme depicted in Figure 2-1.

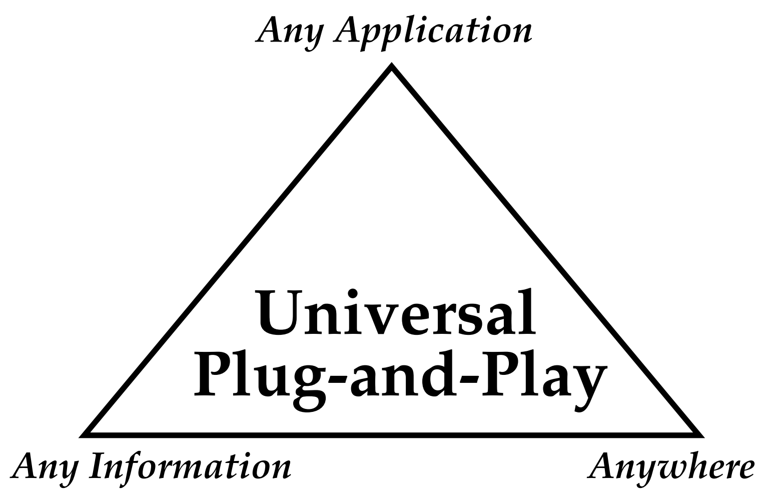


Figure 2-1 Universal Plug-and-Play: The Application Integration Requirement

ToolTalk meets the universal plug-and-play integration requirement:

- It supports message passing between any applications.
- It accesses any information types.
- It integrates workgroups and provides global access while hiding the networking details.
- It makes messaging entirely transparent to users.
- It provides high performance and throughput.

Applications that use the ToolTalk service cooperate by sending and receiving messages. The messages passed by cooperating applications form a small, well-defined message protocol that maximizes both application integration and autonomy. Many applications behave as though they are just a single application.

ToolTalk supports procedural and object-oriented messaging. It leverages the functionality of other applications and fits easily into the mix of procedural applications that dominate current usage.

If message protocols are observed, cooperating applications can be modified — even rewritten — without affecting one another.

Some advantages of the inter-application communication provided by the ToolTalk service are:

- Applications can do the tasks they do best and ask other applications to perform tasks for which they have little or no functionality.
- Inter-application communication is a small investment with the ToolTalk service. Existing applications are easily modified to send, receive, and use shared information.
- No knowledge about the application receiving the information is needed — including its location on the network. The ToolTalk service determines who receives the information by looking for applications who asked for the type of information being sent.
- Role addressing enables addressing to be generic. It describes the functions the addressee needs to perform, such as editing, instead of calling a specific named application. The appropriate application is then addressed.

The ToolTalk Service

The ToolTalk service is one of the first commercial products in the new class of mechanisms designed to facilitate information exchange between applications from different developers. These mechanisms play a key role in application interoperability. Figure 2-2 summarizes how the ToolTalk service fits into a networked environment.

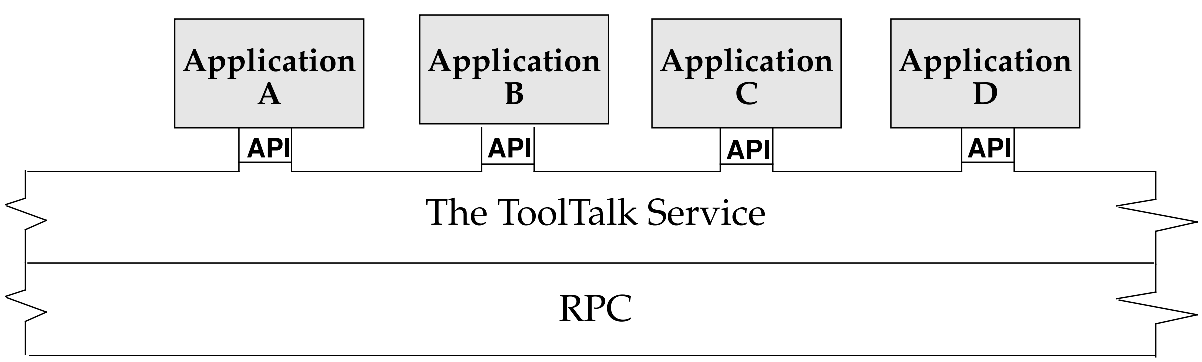


Figure 2-2 Application Interoperability Facilities

The ToolTalk service is used in two ways:

- Applications use the ToolTalk service directly, calling functions from the ToolTalk API library to create, send, and receive messages. Examples of these message include messages to open, display, or close application windows.
- Applications could use a “service” built on top of the ToolTalk service. These application services would then use the ToolTalk service as a communication backbone. These types of services can provide linking and other capabilities for application users.

Both methods provide a unique opportunity for applications to bridge the communication gap and provide cross-application functionality to users.

The ToolTalk service runs on top of the ONC+/TI-RPC product.

Benefits of the ToolTalk Service

Some highlights of the ToolTalk service capabilities are:

- Message addressing is very flexible. A message can be addressed to a specific process, to a particular type of process, or to a process that can perform a particular operation.
- Standard message sets allow developers to communicate between independently developed applications.
- Applications can send notice messages that announce events and request messages that ask for operations to be performed. Multiple processes may receive a copy of either kind of message for observation, but only one process executes a request.
- An application can register interest in receiving messages even though no process is running the application. It can further direct the ToolTalk service to start a process when a message is available, or to save the message until a process running the program is started in another way.

Benefits to Application Developers

The ToolTalk service provides solutions meeting the requirements of multicast messaging primarily for developers of procedural-based, self-contained applications. ToolTalk takes independent applications that solve only portions of the end user’s problem and allows them to interoperate to solve whole problems. Instead of requiring developers to create an entire set of applications that solves a problem, developers can enhance and integrate existing applications to perform the functions needed to solve end users' problems.

By using message sets, little knowledge of other applications is needed to communicate through the ToolTalk service, and developers can concentrate on applications that "plug and play" with other vendor applications. This is particularly important because it means companies do not have to cultivate relationships with every vendor they need to communicate with, nor do they have to change their code every time a new application is added to the set of applications that interoperate.

Benefits to End Users

As applications work together, communicating about events that have occurred or operations that need to be performed, the user's workflow becomes automated. Instead of performing one task with one application and then inputting the results into another application to perform the next task, users focus on solving an entire problem.

When applications communicate, end users gain greater freedom in choosing the application that best suits their needs. Today, a single company can provide a set of applications that may or may not interact. Users choose sets of applications based on trade-offs of the entire set of applications, not on the basis of whether the individual application meets the unique needs for the portion of the problem the application solves. As users gain the freedom to choose applications that interoperate, they can tailor their environment for their unique work situation.

Using the ToolTalk Service

To communicate using the ToolTalk service, applications create, send, and receive ToolTalk messages. Sending applications create, fill in, and send a message; the ToolTalk service determines the recipient and delivers the message to the receiving application.

ToolTalk messages are simple structures that contain fields for address, subject, and delivery information. To send a ToolTalk message, an application obtains an empty message, fills in the message attributes, and sends the message. The sending application needs to provide the following information:

- Is the message a notice or a request? (that is, should the recipient respond to the message?)
- What interest does the recipient share with the sender? (for example, is the recipient running in a specific user session or interested in a specific file?)
- What operation should be performed or has been performed? (that is, are there arguments needed to perform the operation?)

To narrow the focus of the message delivery, the sending application can provide more information in the message.

An important ToolTalk feature is that senders need to know little about the recipients, because applications that want to receive messages explicitly state how these messages should appear. This information is registered with the ToolTalk service in the form of *message patterns*.

Message patterns usually look very similar to the message protocols that applications have agreed to use. An application may add more types of messages they wish to receive beyond the list in the message protocol; for example, an application may want to observe all messages about a particular file.

Applications using the ToolTalk service to communicate usually have something in common — the applications are running in the same user session or they are interested in the same file or data. Applications register this interest in sessions or files, or both, with the ToolTalk service. This file and session information is used in conjunction with the message patterns by the ToolTalk service to determine who should receive a message.

For example, one application wants to inform all interested applications running in a user's session that an event has occurred. Because network window systems, such as the OpenWindows system, allow users to run applications on multiple machines, it is difficult for the sending application to know which applications should be notified and where they are running. Using the ToolTalk library functions, the sending application sends a message scoped to the session, and the ToolTalk communication process delivers copies of the message to all applications in the session that have registered a pattern that matches the message, as illustrated in Figure 2-3.

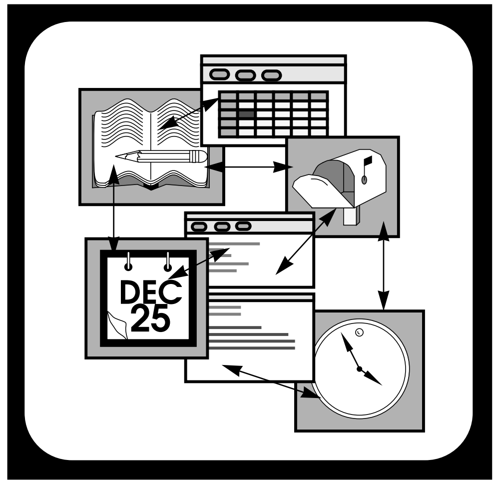


Figure 2-3 ToolTalk Single-User "Session-Scoped" Messaging Between Applications

In file scoping, the candidates for pattern matching are limited to those interested in a particular piece of information that is represented by a file. This type of scoping can integrate many applications that run on behalf of different users where the applications are working on the same information. Figure 2-4 shows file scoping.

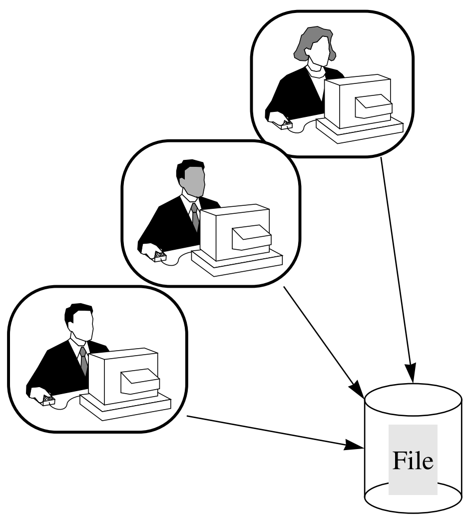


Figure 2-4 ToolTalk Workgroup "File-Scoped" Messaging Between Applications

ToolTalk Messages

Processes use the ToolTalk service to send each other messages. Each message consists of attributes such as those shown in Table 2-1. Within a message's scope, the ToolTalk service determines the actual receivers by matching attributes.

Table 2-1 Basic Message Attributes

Attribute	Value
Address	procedure or process
Class	request or notice
Operation	action to be performed, or event that has occurred
Arguments	input values for operation, output results
Scope	session, file, both, file-in-session
State	created, sent, handled

Sending processes supply values for message attributes. Receiving processes specify, in the message patterns they register, the attribute values of messages they want to receive. The ToolTalk service compares message attributes to receiver patterns, and delivers copies of a message to the processes that have registered matching patterns.

Notices and Requests

Processes can send two classes of ToolTalk messages: notices and requests. A *notice* is informational, a way for a process to announce an event. Processes that receive a notice need not return any results to the sender. A *request* is a call for an action with the results of the action recorded in the message, and the message is returned to the sender as a reply.

Handlers and Observers

For each pattern describing a message it wants to receive, a process also declares whether it can handle the message or only wants to observe it. Both requests and notices can be handled or observed. Although many processes can observe a message, only one process can handle it; thus, a requested operation is performed only once. If the ToolTalk service cannot find a handler for a request message, it returns the message to its sender with a state of *failed*. Because notices do not return results, an unhandled notice is not an error.

Routing

A notice takes a one-way trip, as shown in Figure 2-5. The sender creates a message, fills in attribute values, and sends it. The ToolTalk service matches message and pattern attribute values, and gives a copy of the message to one handler and to all matching observers. File-scoped messages are automatically transferred across session boundaries to processes that have declared interest in the file.

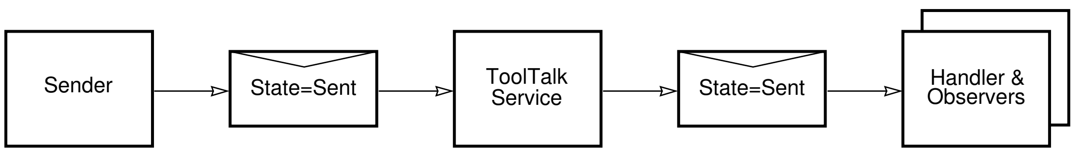


Figure 2-5 Notice Routing

A request, as shown in Figure 2-6, takes a round-trip from sender to handler and back to sender: copies of the message take one-way side trips to observers. The ToolTalk service delivers a request to only one handler. The handler adds results to the message and sends it back. Other processes can observe a request before it is handled, after it has been handled, or at both times; observers absorb a request without sending it back.

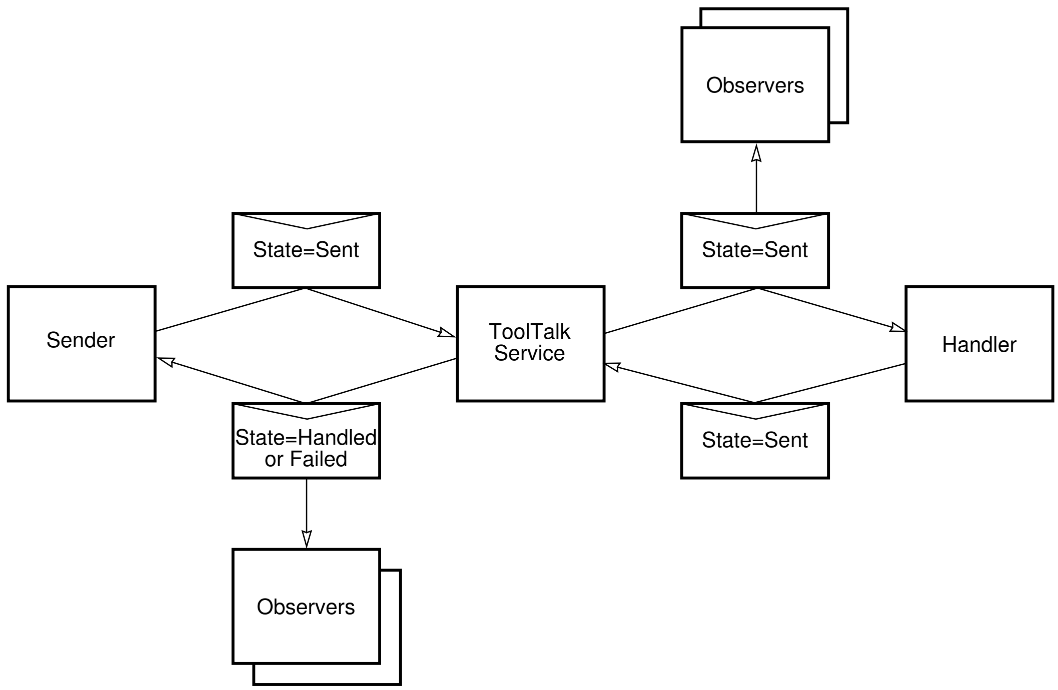


Figure 2-6 Request Routing

Addressing

Although messages are ultimately delivered to processes, they can be addressed to processes or procedures. Each ToolTalk process that receives messages has a process identifier (`procid`) that is unique across all running sessions. A sender that knows a receiver's `procid` can send a message to that process.

However, one process does not usually know another process's `procid`; more often a sender does not care which process executes an operation (request message) or learns of an event (notice message). A procedure message is received by any process that has registered a matching pattern. The sender specifies an operation and arguments, and lets the ToolTalk service finds one handler and all matching observers.

Standard Message Sets

Applications usually are not developed concurrently. And typically they are developed by multiple groups. Users often want to substitute their favorite applications, so they want the universal plug-and-play across all applications. ToolTalk's standard message sets provide integration benefits to users of compliant applications.

By having messages organized into discrete sets, you get a pre-arranged interface between applications with common interests. These sets are specific to an application domain; for example, accounting or computer-aided software engineering (CASE). Figure 2-7 represents the relationships among the message sets.

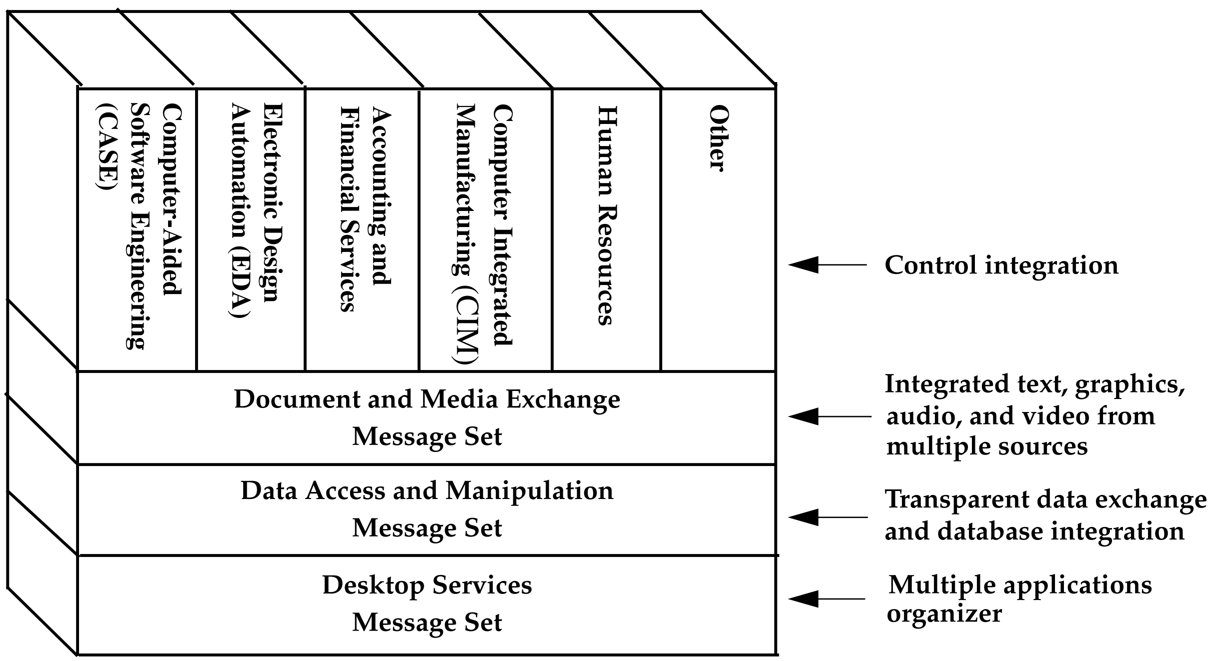


Figure 2-7 Standard ToolTalk Message Sets and Integration Benefit Examples

As the result of extensive work to define message standards with leading software suppliers and end users, SunSoft provides these well-defined, highly functional, standard message sets to anyone. They are open specifications in the public domain without royalty or license fees.

ToolTalk-compliant applications realize many integration benefits from the various message sets, which are described below.

Desktop Services Message Set

To achieve basic desktop integration, applications need to support a basic set of messages to enable inter-application control. With the Desktop Services Message Set, an application can manage the state of files, windowing, and processing for itself or any other application. A powerful messaging protocol that benefits both developers and users of desktop applications, the Desktop Services Message Set allows applications to interact easily with other desktop applications.

For integrated control of applications to take place, certain basic features are needed to launch, halt, control display appearance, and pass information regarding input and output data. All applications need to have these facilities so that other applications in the toolset can inter-change basic control information. This kind of functionality enables the development of smart desktops and integrated smart toolsets. Groups of applications can now call upon one another to perform tasks and to interact as one solution environment for the end user.

Data Access and Manipulation Message Set

Users who access databases and manipulate information in spreadsheets and decision support applications need transparent integration. The Data Access and Manipulation Message Set makes data integration seamless and transparent to users.

Document and Media Exchange Message Set

No single vendor provides a completely integrated solution that meets the complex needs of today's market. The ToolTalk Document and Media Exchange Message Set allows any application to incorporate text, graphics, audio, and video generated by another application. Virtually all applications can benefit from close integration of documentation and multimedia.

Using ToolTalk Document and Media Exchange Message Set, multimedia applications can communicate with one another in a transparent manner, both locally and over networks, regardless of data formats, compression technology, and other technical issues, all of which have previously confined the use of this technology.

CASE Interoperability Message Sets

To achieve control flow integration in an integrated CASE environment, applications need to support messages that enable inter-application control. The CASE Interoperability Message Sets are the common message sets that provide this functionality for CASE applications. A powerful messaging definition that benefits both developers and users of CASE applications, the CASE Interoperability Message Sets allow applications to interact easily with other CASE applications. The CASE Interoperability Message Sets define a standard set of CASE functionality available through control integration services. Each message set describes an area of functionality in the CASE environment. The message sets are:

- Analysis and Design
- Browsing
- Building
- Debugging
- Editing
- Version Management

Adopting Message Sets

It is easy to adopt ToolTalk messages. With help from a series of technical white papers, an experienced programmer can learn how to incorporate them into new or existing applications.

If you are a software vendor, you are not obligated to adopt all message sets. You can use some or all, depending on the functional integration needs of your customers. Your applications can then interoperate with any other application with the same message set(s).

Moving Toward Distributed Objects

ToolTalk is a part of SunSoft's Project DOE (Distributed Objects Everywhere). Through Project DOE, SunSoft has a major strategic commitment to deliver the power of the distributed object paradigm with an evolutionary growth path for current applications. ToolTalk can be used today, and ToolTalk message compliant applications will evolve smoothly into the future.

Typically, the procedural and object applications of today group together large collections of functionality and are known as "large-grained applications" [see Figure 2-8]. The ToolTalk service integrates these applications through multicast messaging.

As SunSoft delivers to market the Project DOE system facilities that enable distributed objects, developers will be able to create new kinds of applications based on the distributed object model. This finer grained application construction model will allow the "plug-and-play" construction of applications from distributed object components.

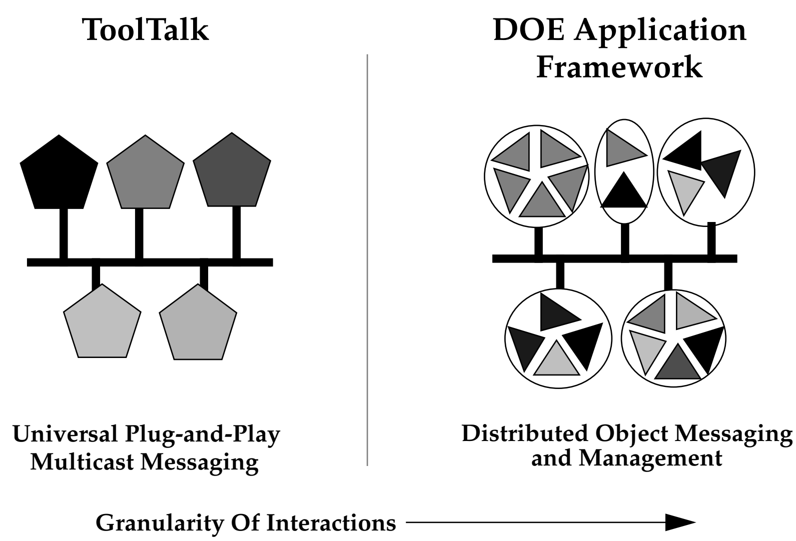


Figure 2-8 ToolTalk's Large-Grained Message Integration Leads to Finer-Grained Distributed Object Integration

SunSoft will move the ToolTalk messaging API into the Application Framework and will support all current ToolTalk functionality. Figure 2-9 shows how an API migration to DOE protects your investment in ToolTalk messages.

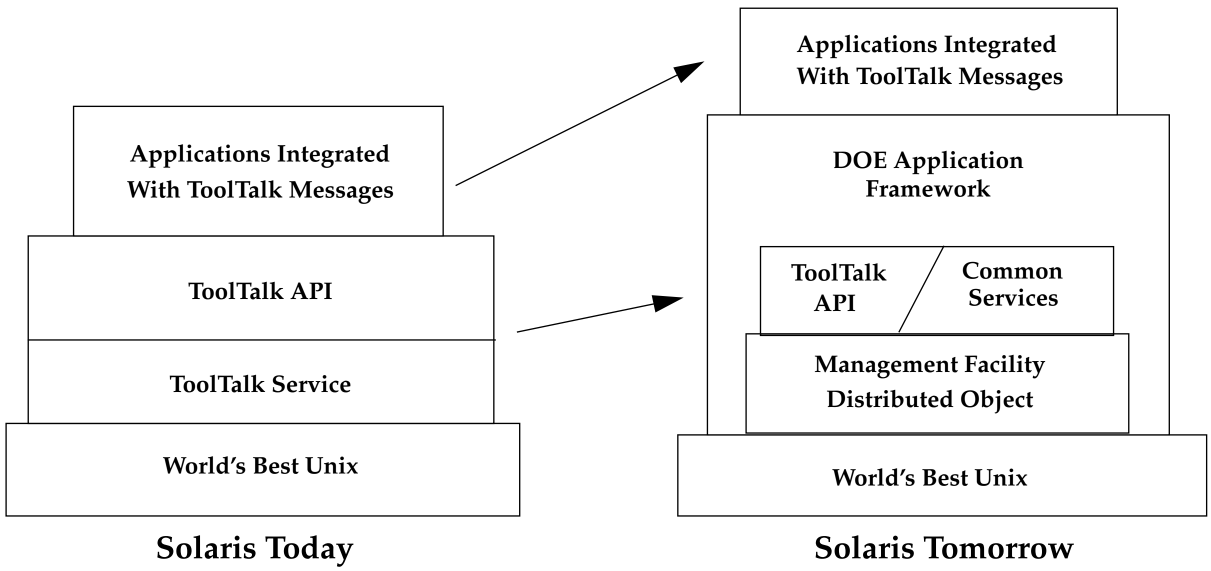


Figure 2-9 API Migration to DOE

From a ToolTalk functionality perspective, this move will be completely transparent to users and developers. Over time, the underlying ToolTalk implementation that users and developers do not see will be swapped out and replaced by the Distributed Object Management Facility (or DOMF) and certain common services.

This is great for developers and users, because it preserves current application functionality while providing access to distributed solutions in the future. Investments made today to integrate existing large-grained applications via ToolTalk will pay off today through increased user productivity. These investments will also prepare you for your evolution into distributed object applications.

Today there is a need for applications to integrate automatically with other applications. By cooperatively using one another's facilities, interoperating applications offer users capabilities that would be difficult to provide in a single application. The ToolTalk service is designed to facilitate the development of interoperating applications that serve individuals and workgroups.

The ToolTalk service is the solution to the integration problem. It is enabling technology that provides the following:

- It supports the *universal plug-and-play* requirements of today's computer users. They gain greater freedom in choosing applications that interoperate with one another.
- It provides multicast messaging solutions to meet the integration requirements of developers of procedural-based, self-contained applications.
- Its standard message sets provide a pre-arranged interface between applications with common interests.
- It includes a transition path from today's procedure-oriented applications to distributed object applications of the future.
- It is a standard feature of the Solaris 1.*x* and Solaris 2.*x* operating environments. It is available at no extra cost.

SunOS 3

Swap Meet[1]

By – Hal Stern, SMCC, Northeast Area

Deciding how much swap space you need on a given machine is a black art. Rules of thumb like "three times physical memory" are popular folklore, but they don't apply to machines with very large memories. Swap spaces upwards of a gigabyte don't sit well with many system administrators, or those managers who need to purchase disks used solely for swap space.

You need to plan for both average swap space usage and peak demands. Few things are as frustrating as not being able to start the last few processes that clean up after everything else, due to a lack of swap space. The seemingly simple task of allocating swap space is made complex by the SunOS 4.*x* virtual memory (VM) system and the different answers given by utilities that purportedly measure swap space usage. To add another spin, swap space allocation rules change again in Solaris 2.0.

1. Originally published in *SunWorld*, January 1993, pp. 89-92. Copyright © 1992 by Integrated Media, Inc. Reprinted with permission from *SunWorld* magazine.

An obvious starting place is to list those things that consume swap space. We'll then decipher what various SunOS utilities tell you about swap space usage, and present some metrics for estimated future needs. Finally, a description of the new swap filesystem (`swapfs`) in Solaris 2.0 should make you believe that there *will* be simplicity in managing large memory configurations.

What's in a Name?

SunOS processes contain two kinds of address space segments: those that are mapped in from the filesystem, and *anonymous* segments that have no corresponding file. Examples of the former include the text segment of a process, shared libraries, and files that are accessed via `mmap()`. Anonymous segments include the data segment, stack, dynamically allocated storage, System V shared memory segments, and pages resulting from copy-on-write operations. Mapped segments can be paged right to or from the filesystem, but anonymous memory segments don't have a named entry in the filesystem. All anonymous memory segments use the swap device as "backing store" for paging operations.

Pretty simply, anonymous memory is the lion's share of the memory used by a process. If you have a 10,000-line C program that operates on a 100 Mbyte data array, the mapped data segments constitute less than 1% of the total address space of the process. You need to allocate enough swap space for most of the total virtual address space used by your processes, unless you rely heavily on `mmap()`. In that case, the swap space requirement is offset by filesystem space consumed by the mapped files.

The SunOS 4.*x* swap usage is a departure from the standard Berkeley UNIX policy, in which the entire process was paged to the swap device, and the text segment was *copied* to the swap device before a process even began execution. In SunOS 4.*x* and later, anything with a filesystem name gets paged directly to the filesystem, defeating the purpose of the BSD "sticky bit." Intended to accelerate the startup of frequently used executables such as `vi` and `ls`, the sticky bit forced the kernel to retain text segment pages on the swap device even after the process using them had exited. The SunOS VM system uses the page cache to retain recently used text pages, and never copies them to the swap device.

Leave Room for Later

A system's total virtual memory in SunOS 4.*x* is the size of the swap space plus the small amount consumed by files mapped directly from the filesystem. Because most of a process is anonymous memory, the amount of swap space really determines the total virtual memory available in the system. In SunOS 4.*x*, a new page of swap space is allocated whenever a page is added to the address space of a running process. Call `malloc()` to get 10 Mbytes of memory and you'll also force 10 Mbytes of swap space to be allocated.

Doing the allocation up front ensures that the system can safely page out the entire process at some later point, when it needs the memory for something else. These swap pages might never be used — your process may run to completion and never be paged out — but the kernel takes the conservative route and ensures that it won't have to deal with problems later on when it goes to page something out to the swap space.

The allocate-before-use policy, though, can be a bit annoying if you have a large memory configuration and know that you'll be running jobs that fit completely within physical memory. You still need to provide a swap area at least equal to the size of your physical memory, or you won't be able to use all of it. As soon as you run out of swap space, attempts to allocate more pages will fail, even if there is unused physical memory in the system. Watch someone purchase memory, install it, and then have a simulation job's memory allocations fail with `ENOMEM` errors. The looks of despair and confusion rival those of kids who don't get a pony for Christmas twenty or thirty years in a row.

Use What You Take

Now that you know what goes into the swap space, you should be able to evaluate how much you're using and estimate how much you'll need for different workloads. The easiest way to view swap usage is with the `pstat` command:

```
% pstat -s
5656k allocated + 688k reserved = 6344k used, 111584k available
```

The `used` figure tells you how much of the swap space is active. Breaking the usage down into `allocated` and `reserved` components gives you a feel for how much space has been withheld by the system for potential future requirements. Creating a new System V shared memory segment or duplicating an address space via `fork()` causes some additional pages to be reserved; they're released when the reserving process exits.

These statistics tell you if you're up against your swap space limit or if you've grossly overestimated what you need. They are only aggregates, however, and won't help you configure that new CPU server sitting on the loading dock. If you want to plan for adding swap space as you add users, or you are planning for a multiuser or X terminal server, you need to come up with a profile for each user's set of processes.

Your best bet is to use the old standard `ps` to look at processes of interest:

```
% ps -u
USER    PID %CPU %MEM   SZ  RSS TT STAT START   TIME  COMMAND
stern  10606  0.0  0.0   76    0 p5 IW   Nov 2  0:00  -bin/csh (csh)
stern  10592  0.0  0.0  428    0 co IW   Nov 2  1:30  stktool -Wp 219 1035 -Ws
stern  10806  0.0  4.3  540 1292 p6 S    Nov 2  0:18  123.exe
stern  10657  0.0  0.0 3280    0 p5 IW   Nov 2  3:42  maker
```

Look at the `SZ` field: it tells you how much memory the stack and data segments are using. This is a reasonable estimate of the amount of swap space used by the processes; sum the `SZ` fields to determine a lower bound on the amount of swap space needed to support this workload.

A simple `awk` or `perl` script will do the tallying for you:

```
% ps -u | sed -e '1d' | awk '{total += $5 } END { print total }'
4324
```

The header from `ps` is removed by `sed`, and `awk` then sums the values in the `SZ` column. In this example, 4.3 Mbytes of swap are used by the four processes owned by "`stern`". Note that the `SZ` column reports how much anonymous memory you have used, while the `RSS` column tells you how much of the total process — including mapped segments — is loaded into memory at this time. Those processes with an `RSS` of zero have been swapped out.

It's important to note that the workload estimates produced from `ps` output are only lower bounds. Add up the `SZ` fields for all processes and you'll come up short compared to what `pstat -s` tells you. The `SZ` figures reported by `ps` don't include pages resulting from copy-on-write operations, the user area, or shared memory segments using swap space. Be sure to pad your estimates when configuring a machine, using lower bounds determined by this kind of workload evaluation.

Dealing With No-Shows

Every popular restaurant has to deal with no-shows: people make reservations and then fail to use the space allocated for them. The same is true of swap space allocations. The system reserves pages of swap space when a process requests the new virtual pages, but the swap pages may never be used. When you're running a large memory job that is guaranteed to be memory-resident, all of the swap space allocations are for naught, and leave you with a large disk that's reserved but unused.

Enter `swapfs` in Solaris 2.0. The basic idea of `swapfs` is to create virtual swap space that isn't mapped to disk blocks until it is used for a page-out operation. In our restaurant example, think of `swapfs` as a second reservation book in which every telephone reservation request is recorded. When you finally show up at the door, your reservation is copied to the "real" reservation book and space can be allocated for you.

From the system administrator's point of view, `swapfs` changes the definition of total virtual memory from "swap space plus mapped files" to "swap space plus most of physical memory." Some physical memory used for the kernel and `swapfs` itself is removed from the pool managed by `swapfs`. Use the Solaris 2.0 `swap` command to examine the swap devices:

```
% /usr/sbin/swap -l
swapfile             dev    swaplo  blocks   free
swapfs                -          0  187488   183040
/dev/dsk/c0t3d0s1    32,25       8  131752   123128
```

The disk device is a normal "`b`" partition swap device. Note that the pool managed by `swapfs` is about 60k blocks larger than the swap device — accounting for most of the 32 Mbytes of memory on this machine.

The main goal of `swapfs` is to avoid having to allocate swap space that is never used. This obviously changes your swap allocation policy. Instead of calculating the total virtual memory used from the stack and data segment sizes, figure out how much virtual memory you'll use in excess of the physical memory on the machine. Only this overflow must be backed by swap space under Solaris 2.0. If you can keep all of your processes memory-resident, then you'll only need a small swap space of about 30 Mbytes to provide backing store for the standard system processes.

If you’re planning an upgrade to Solaris 2.0, or installing new memory in a large server, plan out how you’ll deal with the swap requirements before you start. It’s the easiest way to prevent people from coming to you, looking for “lost” memory they’re sure they installed.

Security 4

Solaris Basic Security Module (BSM)

This article is excerpted from *Solaris Basic Security Module (BSM)* (part number 800-8933-10). For a full technical description of the BSM product and how to install and administer it, please refer to the official document in its entirety.

Introduction

The Solaris Basic Security Module (BSM) provides the security features required by the C2 Class of the Trusted Computer System Evaluation Criteria (TCSEC) that are not included in the standard Solaris 2.1 release. The additional features are the security auditing subsystem and a device allocation mechanism that provides the required object reuse characteristics for removable or assignable devices. C2 discretionary access control and identification and authentication features are provided by the standard Solaris 2.1 system.

Administering Auditing

This section describes how to set up and administer auditing. Auditing makes users accountable for their actions. The auditing mechanism allows an administrator to detect potential security breaches. Auditing can reveal suspicious or abnormal patterns of system usage and provide the means to trace suspect actions back to a specific user. Auditing may serve as a deterrent: if users know that their actions are likely to be audited, they may be less likely to attempt malicious activities.

Successful auditing depends on two other security features: *Identification* and *Authentication*. At login, after a user supplies a User Name and Password, a unique audit ID is associated with the user's process. The audit ID is inherited by every process started during the login session. Even if a user changes identity (by using `su(1)`), all actions performed are tracked with the same audit ID.

Auditing makes it possible to:

- monitor security-relevant events that take place on the system
- record the events in an *audit trail*
- detect misuse or unauthorized activity (by analyzing the audit trail)

During system configuration, the system administrator selects which activities to monitor. The administrator may also fine-tune the degree of auditing that is done for individual users.

After audit data is collected, audit reduction and interpretation tools allow the examination of interesting parts of the audit trail. For example, you may choose to look at audit records for individual users or groups, look at all records for a certain type of event on a specific day, or select records that were generated at a certain time of day.

Audit Startup

Auditing is enabled by starting up the audit daemon, `auditd(1M)`. This can be done manually be executing `/usr/sbin/auditd` as the superuser (`root`).

The existence of a file with the pathname `/etc/security/audit_startup` causes the audit daemon to be run automatically at system startup time. The `audit_startup(4)` file is actually an executable script that is invoked as part of the startup sequence just prior to the execution of the audit daemon. A default `audit_startup(4)` script is set up during the BSM package installation that automatically configures the event to class mappings and sets the audit policies.

Audit Classes and Events

Security-relevant actions may be audited. The system actions that are auditable are defined as *audit events* in the /usr/security/audit_event file. Each auditable event is defined in the audit_event(4) file by a symbolic name, an event number, a set of preselection classes, and a short description.

Most actions (events) are attributable to an individual user. However, some events are non-attributable because they occur before a user is identified and authenticated, or they occur at the kernel interrupt level.

Each audit event is also defined as belonging to an *audit class* or classes. The mapping of audit events to classes is configurable. The entries in the audit_event(4) file assign the event to one or more classes in the third column.

Whether or not an auditable event is recorded in the audit trail depends on whether the administrator preselects a class for auditing that includes the specific event. 32 actual audit classes are defined, besides the two global classes: all and no.

Kernel Events

Events generated by the kernel (system calls) have event numbers between 1 and 2047. The event names for kernel events begin with AUE_ , followed by an UPPERCASE mnemonic for the event. For example, the event number for the creat(2) system call is 4 and the event name is AUE_CREAT .

User-level Events

Events generated by application software outside the kernel are numbered greater than 6144. The event names begin with AUE_ , followed by a lowercase mnemonic for the event.

Audit Records

Each *audit record* describes the occurrence of a single audited event, including such information as who did the action, which files were affected, what action was attempted, and where and when it occurred.

The type of information saved for each audit event is defined as a set of *audit tokens*. Each time an audit record is created for an event, the record contains some or all of the tokens defined for it, depending on the nature of the event.[1]

When the `praudit` command creates human-readable versions of the audit records, the short description of the event is included in the default output of `praudit`; the symbolic name is included in the output of `praudit -s`; and the event number is included in the output of `praudit -r`.[2]

Audit Flags

Audit *flags* indicate classes of events to audit. Machine-wide defaults for auditing are specified for all users on each machine using flags in the `audit_control(4)` file, which is described in the section below, "The `audit_control` File."

The system administrator can modify what gets audited for individual users by putting audit flags in a user's entry in the `/etc/security/audit_user` file. The audit flags are also used as arguments to the `auditconfig` command.

Definitions of Audit Flags

Each audit class is shown in Table 4-1 with the audit flag (which is the Short Name that stands for the class), the long name, and a short description. The system administrator uses the audit flags in the auditing configuration files to specify which classes of events to audit.

1. The audit record descriptions in Appendix A of *Solaris Basic Security Module (BSM)* list all the audit tokens defined for each event and what each token means.

2. Two cross-reference tables in Appendix A of *Solaris Basic Security Module (BSM)* translate the event numbers and symbolic names to event descriptions to help you interpret the output of `praudit`. How the `praudit(1M)` command reads and interprets audit records is described in Chapter 4 of the *BSM* manual ("Audit Trail Analysis") and in the `praudit(1M)` man page.

Table 4-1 Audit Classes

Short Name	Long Name	Short Description
`no`	invalid class	Null value for turning off event preselection
`fr`	file read	Read of data, open for reading, etc.
`fw`	file write	Write of data, open for writing, etc.
`fa`	file attribute access	Access of object attributes: `stat`, `pathconf`, etc.
`fm`	file attribute modify	Change of object attributes: `chown`, `flock`, etc.
`fc`	file create	Creation of object
`fd`	file delete	Deletion of object
`cl`	file close	`close(2)` system call
`pc`	process	Process operations: `fork`, `exec`, `exit`, etc.
`nt`	network	Network events: `bind`, `connect`, `accept`, etc.
`ip`	ipc	System V IPC operations
`na`	non-attribute	Non-attributable events
`ad`	administrative	Administrative actions
`lo`	login or logout	Login and logout events
`io`	ioctl	`ioctl(2)` system call
`ex`	exec	Program execution
`ot`	other	Miscellaneous
`all`	all	All flags set

Audit Flag Syntax

Depending on the prefixes, a class of events can be audited whether it succeeds or fails, or only if it succeeds or only if it fails. The format of the audit flag is shown here.

<*prefix*><*flag*>

Table 4-2 shows prefixes that specify whether the audit class is audited for success or failure or both.

Table 4-2 Prefixes Used in Audit Flags

Prefix	Definition
none	audit for both success and failure
+	audit for success only
–	audit for failure only

To give an example of how these work together, the audit flag `lo` means all successful attempts to log in and log out and all failed attempts to log in (you cannot fail an attempt to logout). For another example, the `-all` flag refers to all failed attempts of any kind, and the `+all` flag refers to all successful attempts of any kind.

Caution – The `+all` flag generates large amounts of data that can fill up audit filesystems quickly, so use it only if you have extraordinary reasons to audit everything.

Prefixes to Modify Previously-Set Audit Flags

Use the following prefixes in either of two ways: in the `flags:` line in the `audit_control` file to modify already-specified flags, and in flags in the user's entry in the `/etc/security/audit_user` file.

The prefixes in Table 4-3, used along with the short names of audit classes, turn on or turn off previously-specified audit classes. These prefixes turn on or off previously-specified flags only.

Table 4-3 Prefixes Used to Modify Already-Specified Audit Flags

Prefix	Definition
`^-`	Turn off for failed attempts
`^+`	Turn off for successful attempts
`^`	Turn off for both failed and successful attempts

The `^-` prefix is used in the `flags` line in the following example from an `audit_control` file.

In the example below, the `lo`, `nt`, and `ad` flags specify that all logins, network, and administrative operations are to be audited when they succeed and when they fail. The `-all` means audit all failed events. Because the `^-` prefix means turn off auditing for the specified class for failed attempts, the `^-fc` flag modifies the previous flag that specified auditing of all failed events; the two fields together mean "audit all failed events, except failed attempts to create filesystem objects."

```
flags:lo,nt,ad,-all,^-fc
```

The audit_control *File*

An audit_control(4) file on each machine is read by the audit daemon, auditd(1M). The audit_control file is located in the /etc/security directory. A separate audit_control file is maintained on each machine, because machines in the distributed system may be mounting their audit filesystems from different locations or specifying them in a different order. For example, the primary audit filesystem for ***machine_a*** might be the secondary audit filesystem for ***machine_b***.

The system administrator specifies four kinds of information in four kinds of lines in the audit_control file:

- The *audit flags* line (flags:) contains the audit flags that define what classes of events are audited for all users on the machine. The audit flags specified here are referred to as the *machine-wide audit flags* or the *machine-wide audit preselection mask*. Audit flags are separated by commas, with no spaces.
- The *non-attributable flags* line (naflags:) contains the audit flags that define what classes of events are audited when an action cannot be attributed to a specific user. The flags are separated by commas, with no spaces.
- The *audit threshold* line (minfree:) defines the minimum free space level for all audit filesystems.

  The minfree percentage must be greater than or equal to 0. The default is 20%.
- The *directory definition* lines (dir:) define which audit filesystems and directories the machine will use to store its audit trail files.

  There may be one or more directory definition lines. The order of the dir: lines is significant, because auditd opens audit files in the directories in the order specified. The first audit directory specified is the primary audit directory for the machine, the second is the secondary audit directory where the audit daemon puts audit trail files when the first one fills, and so forth.

The administrator creates an audit_control file during the configuration process on each machine.

After the audit_control file is created during system configuration, the administrator may later edit the file. After any change, the administrator enters audit -s to instruct the audit daemon to reread the audit_control file.

Sample audit_control *File*

Following is a sample audit_control file for the machine ***dopey***. ***dopey*** uses two audit filesystems on the audit server ***blinken***, and a third audit filesystem mounted from the second audit server ***winken***, which is used only when the audit file system on ***blinken*** fills up or is unavailable. The minfree percentage of 20% specifies that the warning script (/etc/security/audit_warn) is run when the filesystems are 80% filled and the audit data for the current machine will be stored in the available audit directory, if any. The flags specify that all logins, network, and administrative operations are to be audited (whether or not they succeed), and that failures of all types are to be audited except failures to create a filesystem object.

```
flags:lo,nt,ad,-all,^-fc
naflags:lo,nt
minfree:20
dir:/etc/security/audit/blinken/files
dir:/etc/security/audit/blinken.1/files
#
# Audit filesystem used when blinken fills up
#
dir: /etc/security/audit/winken
```

User Audit Fields in th e audit_user *File*

If it is desirable to audit some users differently from others, the administrator may edit the audit_user file to add audit flags for individual users. If specified, these flags are combined with the system-wide flags specified in the audit control file to determine which classes of events to audit for that user. The flags the administrator enters into the user's entry in the audit_user file modify the defaults from the audit_control file in two ways: by specifying a set of event classes that are never to be audited for this user, or by specifying a set of event classes that are always to be audited.

In the audit_user file entry for each user, there are three fields. The first field is the username; the second field is the *always audit* field; the third is the *never audit* field.

The two auditing fields are processed in sequence, so auditing is enabled by the first field and turned off by the second.

Note – Avoid the common mistake of leaving the `all` set in the *never audit* field. This causes all auditing to be turned off for that user, overriding the flags set in the *always audit* field.

Note – Successful events and failed events are treated separately, so a process can (for example) generate more audit records when an error occurs (such as a security violation) than when the event is successful.

Using the *never audit* flags for a user is not the same as removing classes from the *always audit* set. For example, suppose (as shown in the example below) we have a user named `fred`, for whom we want to audit everything except successful reads of filesystem objects (this is a good way to audit almost everything for a user while generating only about three-quarters of the audit data that would be produced if all data reads were also audited). Here are two possible `audit_user` entries, but only the first one is correct:

```
fred:all,^+fr:
```

```
fred:all:+fr
```

The first one says "always audit everything except successful file-reads." The second one says "always audit everything, but never audit successful file-reads." The second one is incorrect because it overrides the system default. The first example achieves the desired effect: any earlier default applies, as well as what's specified in the `audit_user` entry.

Process Audit Characteristics

The following audit characteristics are set at initial login:

- Process preselection mask
- Audit ID
- Audit Session ID
- Terminal ID (port ID, machine ID)

Process Preselection Mask

When a user logs in, `login` combines the machine-wide audit flags from the `audit_control(4)` file with the user-specific audit flags (if any) from the `audit_user(4)` file, to establish the *process preselection mask* for the user's processes. The process preselection mask specifies whether events in each audit event class are to generate audit records.

The algorithm for obtaining the process preselection mask is as follows. The audit flags from the `flags:` line in the `audit_control` file are added to the flags from the *always audit* field in the user's entry in the `audit_user` file. The flags from the *never audit* field from the user's entry in the `audit_user` file are then subtracted from the total.

(`flags:` line + *always audit* flags) – *never audit* flags =
user's process preselection mask

Audit ID

A process also acquires its audit ID when the user logs in, and this audit ID is inherited by all child processes started by the user's initial process. The audit ID helps enforce accountability. Even after a user becomes `root`, the audit ID remains the same. The audit ID that is saved in each audit record allows the administrator to always trace actions back to a real user.

Audit Session ID

The audit session ID is assigned at login and inherited by all descendant processes.

Terminal ID

The terminal ID consists of the hostname and the Internet address, followed by a unique number that identifies the physical device on which the user logged in. Most of the time, the login will be through the console and the number that corresponds to the console device is 0.

How the Audit Trail is Created

The *audit trail* is created by the audit daemon, `auditd(1M)`. The audit daemon starts running on each machine when the machine is brought up. After the `auditd` program is started at boot time, it is responsible for collecting the audit trail data and writing the audit records into *audit files*, which are also called *audit log files*. [See also the `audit.log(4)` man page for a description of the file format.]

The audit daemon runs as root. All files it creates are owned by root. Even when no auditing has been turned on, auditd continuously operates, looking for a place to put audit records. The auditd operations continue even if the rest of the machine's activities are suspended because the kernel's audit buffers are full. The audit operations can continue because auditd is not audited.

Only one audit daemon may run at a time. An attempt to start a second one will result in an error message, and the new one will exit. If there is a problem with the audit daemon, the administrator should try using audit -t to terminate auditd gracefully and then restart it manually.

The audit_warn(1M) script is run by auditd whenever the daemon switches audit files or encounters difficulty (such as a lack of storage). As distributed, the audit_warn script sends mail to an audit_warn alias and sends a message to the console. Your site should customize audit_warn to suit your needs (described in the section below on "The audit_warn Script"). You must also make sure that the audit_warn alias sends mail to a normal user.

The audit_data *File*

When auditd(1M) starts on each machine, it creates the file /etc/security/audit_data. The format of the audit_data(4) file consists of a single entry with the two fields separated by a colon. The first field is the audit daemon's process ID, and the second field is the pathname of the audit file to which the audit daemon is currently writing audit records. Here is an example:

```
# vi /etc/security/audit_data
116:/etc/security/audit/blinken.1/files/19910320100002.not_terminated.lazy
```

The Audit Daemon's Role

The following list summarizes what the audit daemon, auditd(1M), does:

- It opens and closes audit log files in the directories specified in the audit_control file, in the order in which they are specified.
- It reads audit data from the kernel and writes it to an audit file.
- It sends warnings to the audit_warn alias and to the console when audit directories fill past limits specified in the audit_control file.

- When all audit directories are full, processes that generate audit records are suspended and auditd writes a message to the console and to the audit_warn alias. At this point, only the system administrator could log in to write audit files to tape, delete audit files from the system, or do other cleanup.

When the audit daemon starts up when the machine is brought up to multiuser mode, or when the audit daemon is instructed by the audit -s command to reread the file after the file has been edited, auditd reads the list of directories from the audit_control file and uses those as possible locations for creating audit files.

The audit daemon maintains a pointer into this list of directories, starting with the first. Every time the audit daemon needs to create an audit file, it puts the file into the first available directory in the list, starting at the audit daemon's current pointer. The pointer may be reset to the beginning of the list if the administrator enters the audit -s command. When the audit -n command is used by the administrator to instruct the daemon to switch to a new audit file, the new file is created in the same directory as the current file.

What Makes a Directory Suitable

A directory is *suitable* to the audit daemon if it is accessible to the audit daemon, which means that it must be mounted, that the network connection (if remote) permits successful access, and that the permissions on the directory allow access. Also, in order for a directory to be suitable for audit files, it must have sufficient free space remaining. The administrator may edit the minfree: line in the audit_control file to change the default of 20%. To give an example of how the minfree percentage is applied, if the default minimum free space of 20% is accepted, an e-mail notice is sent to the audit_warn alias whenever a filesystem becomes more than 80% full.

When no directories on the list have enough free space left, the daemon starts over from the beginning of the list and picks the first accessible directory that has any space available until the hard limit is reached. If no directories are suitable, the daemon stops processing audit records, and they accumulate within the kernel until all processes generating audit records are suspended.

Keeping Audit Files Manageable

To keep audit files at a manageable size, a cron job can be set up that periodically switches audit files. Intervals might range from once per hour to twice per day, depending on the amount of audit data being collected.

The audit_warn *Script*

Whenever the audit daemon encounters an unusual condition while writing audit records, it invokes the /usr/etc/audit_warn script. This script can be customized by your site to warn of conditions that might require manual intervention, or to handle them automatically. For all error conditions, audit_warn(1M) writes a message to the console and sends a message to the audit_warn alias. This alias should be set up by the administrator after the BSM package is installed.

When the following conditions are detected by the audit daemon, it invokes audit_warn [see also the audit_warn(1M) man page]:

- An audit directory has become more full than the minfree value allows.

  (The minfree or soft limit is a percentage of the space available on an audit directory.)

  The audit_warn script is invoked with the string soft and the name of the directory as arguments. The audit daemon switches automatically to the next suitable directory, and writes the audit files there until this new directory reaches its minfree limit. The audit daemon then goes to each of the remaining directories in the order listed in audit_control, and writes audit records until each is at its minfree limit.

- All the audit directories are more full than the minfree threshold.

  The audit_warn script is invoked with the string allsoft as an argument. A message is written to the console and e-mail is sent to the audit_warn alias.

  When all audit directories listed in audit_control are at their minfree limits, the audit daemon switches back to the first one, and writes audit records until the directory completely fills.

- An audit directory has become completely full with no space remaining.

  The audit_warn script is invoked with the string hard and the name of the directory as arguments. A message is written to the console and e-mail is sent to the audit_warn alias.

  The audit daemon switches automatically to the next suitable directory with any space available, if any. The audit daemon goes to each of the remaining directories in the order listed in audit_control, and writes audit records until each is full.

- All the audit directories are completely full.

  The audit_warn script is invoked with the string allhard as an argument. A message is written to the console and e-mail is sent to the audit_warn alias. The processes generating audit records are suspended. The audit daemon will go into a loop waiting for space to become available, and resume processing audit records when that happens. While audit records are not being processed, no auditable activities take place—every process that attempts to generate an audit record is suspended. This is one reason that you would want to set up the administrator to run without any auditing, so that the administrator can operate without being suspended.

- An internal error occurs: another audit daemon process is already running (string ebusy), a temporary file cannot be used (string tmpfile), the auditsvc system call fails (string auditsvc), or a signal was received during auditing shutdown (string postsigterm).

  E-mail is sent to the audit_warn alias.

- A problem is discovered with the audit_control file's contents.

  By default, e-mail is sent to the audit_warn alias, and a message is sent to the console.

Using the auditreduce *Command*

The auditreduce command merges together audit records from one or more input audit files. You would usually enter this command from the machine on which all the audit trail files for the entire distributed system are mounted.

Without options, auditreduce merges the entire audit trail (all of the audit files in all of the subdirectories in the audit root directory /etc/security/audit) and sends the merged file to standard output.

The praudit command, described in the section below on "Using praudit", makes the records human-readable.

The following are some of the capabilities provided by options to the auditreduce command:

- Give output containing audit records generated only by certain audit flags.
- Show audit records generated by one particular user.
- Collect audit records generated on specific dates.

How `auditreduce` *Helps in a Distributed System*

When multiple machines running Solaris BSM are administered as part of a distributed system, each machine performs auditable events, and each machine writes audit records to its own machine-specific audit file. This simplifies software and is robust in the face of machine failures. However, unless a tool existed to make it easier, you would have to look at every one of the files to determine a particular user's actions.

The `auditreduce(8)` command makes the job of maintaining the whole audit trail practical. Using `auditreduce` (or shell scripts you write yourself to provide a higher-level interface), you can read the logical combination of all audit files in the system as a single audit trail, without regard to how the records were generated or where they are stored.

The `auditreduce` program operates on the audit records produced by the audit daemon. Records from one or more audit files are selected and merged into a single, chronologically ordered output file. The merging and selecting functions of `auditreduce` are logically independent. `auditreduce` selects messages from the input files as the records are read, before the files are merged and written to disk. [Refer to the `auditreduce(8)` man page.]

`auditreduce` *Examples*

This section describes a few common uses of `auditreduce` to analyze and manage data.

Example 1: How to Display the Whole Audit Log

To display the whole audit trail at once, pipe the output of `auditreduce` into `praudit`.

```
# auditreduce | praudit
```

Example 2: How to Print the Whole Audit Log

With a pipe to `lpr`, the output goes to the printer.

```
# auditreduce | praudit | lpr
```

Example 3: How to Display User Activity on a Selected Data

In the following example, the system administrator checks to see when a user named `fred` logged in and logged out on April 13, 1990, by requesting the **`lo`** message class. The short-form date is in the form *yymmdd* (the long form is described on the `auditreduce(8)` man page).

```
# auditreduce –d 900413 –u fred –c lo | praudit
```

Example 4: How to Copy Login/Logout Messages to a Single File

In this example, login/logout messages for a particular day are summarized into a file. The target file is written in a directory other than the normal audit root. This is the command line:

```
# auditreduce –c lo –d 870413 –O /etc/security/audit_summary/logins
```

The **`–O`** option creates an audit file with 14-character timestamps for both start-time and end-time, and the suffix "`logins`":

```
19870413000000.19870413235959.logins
```

Example 5: How to Clean Up a `not_terminated` *Audit File*

Occasionally, if a system crashes while its audit file is still open, or a server becomes inaccessible and forces the machine to switch to a new server, an audit file remains in which the *end-time* remains the string `not_terminated`, even though the file is no longer being used to write audit records. When such a file is found, you can manually verify that the file is no longer in use and clean it up by specifying the name of the file with the correct options. The following example shows the commands:

```
# cd /etc/security/audit/<server>
# auditreduce –O <machine>
19870413120429.not_terminated.<machine>
```

This creates a new audit file with the correct name (both timestamps) and the correct suffix (***machine***, explicitly specified), and copies all the messages into it.

Other Useful `auditreduce` *Options*

`auditreduce` has many additional options described in the man page. Note that the uppercase options select operations or parameters for `files`, and the lowercase options select parameters for `records`. This subsection shows how to make use of two more useful options.

The *date-time* options **-b** and **-a** allow specifying records before or after a particular day/time. A day begins at *yyyymmdd*00:00:00 and ends at *yyyymmdd*23:59:59. The six parameters of a day are: year, month, day, hour, minute, and second. The digits (**19**) of the year are assumed and need not be specified.

If **-a** is not specified, auditreduce defaults to 00:00:00, January 1, 1970. If **-b** is not specified, auditreduce defaults to the current time of day (GMT). The **-d** option selects a particular 24-hour period, as shown in Example 4 on the previous page.

The auditreduce -a command, with the date shown in the following example, sends all audit records created *after* midnight on July 15, 1991 to praudit:

```
# auditreduce -a 91071500:00:00 | praudit
```

The auditreduce -b command, with the same date as shown above, sends all audit records created *before* midnight on July 15, 1991 to praudit:

```
# auditreduce -b 91071500:00:00 | praudit
```

The message type selection for auditreduce (the -m option) accepts either numeric message identifiers or AUE_*xxxxx* codes. Auditreduce rejects an incorrect format, but does not describe the correct format.

Using praudit

The praudit command reads audit records from standard input and displays them on standard output in human-readable form. Usually, the input is either piped from auditreduce or a single audit file. Input may also be produced with cat(1) to concatenate several files, or tail(1) for a current audit file.

praudit can generate three output formats: default, short (-s option), and raw (-r option). By default, output is produced with one token per line. The -l option requests a whole record on each line. The -d option changes the delimiter used between token fields, and between tokens if -l is also specified.

In -s format, the type is the audit event table name for the event (such as AUE_IOCTL), and in -r format, it is the event number (158, in the case of AUE_IOCTL). That is the only distinction between -s and default format. In -r format, all numeric values (user IDs, group IDs, etc.) are displayed numerically (in decimal, except for Internet addresses, which are in hex, and for modes, which are in octal).

Here is the output from praudit for a header token:

```
header,240,1,ioctl(2),es,Tue Sept 1 16:11:44 1992, + 270000 msec
```

And here is the output from praudit -r for the same header token:

```
20,240,1,158,0003,699754304, + 270000 msec
```

It is sometimes useful to manipulate praudit's output as lines of text—for instance, to perform selections that cannot be done with auditreduce. A simple shell script can process the output of praudit. The following example is called praudit_grep:

```
#!/bin/sh
praudit | sed -e '1,2d' -e '$s/^file.*$//' -e 's/^header/^Aheader/' \\
| tr '\\012\\001' '\\002\\012' \\
| grep "$1" \\
| tr '\\002' '\\012'
```

The example script marks the header tokens by prefixing them with Control-A. (Note that the "^A" is a literal Control-A, not the two characters "^" and "A". Prefixing is necessary to distinguish them from the string "header" that might appear as text in praudit's output.) The script then combines all the tokens for a record onto one line while preserving the line breaks as Control-A; runs grep; and restores the original newlines.

Note that in praudit's default output format, each record can always be identified unambiguously as a sequence of tokens (each on a separate line) beginning with a "header" token and ending with a "trailer" token. Each record, therefore, is easily identified and processed with awk, for instance.

Device Allocation

The TCSEC's object reuse requirement for computing systems at C2 level and above is fulfilled by the device allocation mechanism. This section describes what the administrator needs to know about managing devices.

The administrator must decide whether any devices should be allocatable, and if so, which devices should be allocatable, if the defaults are not appropriate for your site's security policy.

Risks Associated with Device Use

For one example of the security risks associated with the use of various I/O devices, consider how cartridge devices are typically used. Often several users share a single tape drive, which may be located in an office or lab away from where an individual user's own machine is located. This means that, after he or she loads a tape into the tape drive, some length of time may elapse before the user can return to the machine to invoke the command that reads or writes data to or from the tape. Then another time lapse occurs before the user is able to return and take the tape out of the drive. Because tape devices are typically accessible to all users, during the time when the tape is unattended an unauthorized user could access or overwrite data on the tape.

The device allocation mechanism makes it possible to assign certain devices to one user at a time, so that the device can only be accessed by that user while it is assigned to that user's name.

The device allocation mechanism ensures the following for tape devices and provides related security services for other allocatable devices:

- Prevents one user from reading a tape just written by another user before the owner of the tape has removed the tape from the tape drive

Components of the Device Allocation Mechanism

The components of the allocation mechanism that you must understand in order to manage device allocation are:

- the `allocate(1M)`, `deallocate(1M)`, `dminfo(1M)` and `list_devices(1M)` commands
- the `device_allocate(4)` file
- the `device_maps(4)` file
- the lock files that must exist for each allocatable device in `/etc/security/dev`
- the changed attributes of the *device-special files* that are associated with each allocatable device
- the "device clean" scripts for each allocatable device

How any user invokes the `allocate(1M)`, `deallocate(1M)`, `dminfo(1M)`, and `list_devices(1M)` commands is described in the subsection below on using `allocate`. [All of the options and other descriptions are defined in the man pages.]

The `device_allocate(4)` file, the `device_maps(5)` file, and the lock files are specific to each machine. The configuration files are not administered as NIS databases because tape drives, floppy drives, and the printers, are all connected to specific machines.

Using the Device Allocation Utilities

This section describes what the administrator can do with the options to `allocate`, `deallocate`, and `list_devices` that are usable only by the superuser (`root`). The commands are detailed on their respective man pages.

`allocate(1M)`

`-F` ***device_special_filename***
Reallocates the specified device. This option is often used with the `-U` option to reallocate the specified device to the specified user. Without the `-U` option, the device is allocated to `root`.

`-U` ***username***
Causes the device to be allocated to the user specified rather than to the current user. This option allows you to allocate a device for another user while you are `root`, without having to assume that user's identity.

`deallocate(1M)`

`-F` ***device_special_filename***

Devices that a user has allocated are not automatically deallocated when the process terminates or when the user logs out. When a user forgets to deallocate a tape drive, you can force deallocation using the `-F` option while you are `root`.

`-I`

Forces deallocation of all allocatable devices. This option should only be used at system initialization.

`list_devices(1M)`

The administrator can run `list_devices` to get a listing of all the device-special files that are associated with any device listed in the `device_maps` file.

`-U` ***username***

List the devices that are allocatable to the user ID associated with the specified username. This allows you to check which devices are allocatable to another user while you are `root`.

The Allocate Error State

The allocate error state is mentioned in the man pages for the `allocate` components. An allocatable device is in the *allocate error state* if it is owned by user `allocate` and group `wheel` with a device-special file mode of `0100`. If a user wishes to allocate a device that is in the allocate error state, the administrator should try to force the deallocation of the device, using the `deallocate` command with the `-F` option, or use `allocate -U` to assign it to the user, then investigate any error messages that display. When the problems with the device are corrected, the administrator must rerun the `deallocate -F` or `allocate -F` commands to clear the allocate error state from the device.

The device_maps File

You can look at the device_maps(4) file to find out device names, device types, and the device-special files that are associated with each allocatable device. The default device_maps file is created during installation, and it has one record for each allocatable device in the system.

This file defines the device-special file mappings for each device, which in many cases is not intuitive. This file allows various programs to discover which device-special files map to which devices. You can use the dminfo(1M) command, for example, to get the device name, the device type, and the device-special files to specify when setting up an allocatable device; dminfo(1M) uses the device_maps(4) file.

Each device is represented by a one line entry of the form:

device-name **:** ***device-type*** **:** ***device-list***

Lines in device_maps(4) can end with a '\' to continue an entry on the next line. Comments may also be included. A '#' makes a comment of all further text until the next NEWLINE not immediately preceded by a '\'.

Leading and trailing blanks are allowed in any of the fields.

device-name

The name of the device (for example, st0, fd0, audio). You can get the device type for any device from the directly from the device type field in the device_maps file, or use the dminfo(1M) command. (The device name specified here must be used for the name of the lock file for the device.)

device-type

The generic device type (the name for the class of devices, such as st, fd, audio) — used to group related devices. You can get the device type from the device type field in the device's entry in the device_maps file.

device-list

A list of the device-special files associated with the physical device.

For an example of entries for SCSI tape `st0` and floppy disk `fd0` in a `device_maps` file, see the following example:

```
fd0:\
    fd:\
    /dev/fd0 /dev/fd0a /dev/fd0b /dev/fd0c /dev/rfd0 /dev/rfd0a /dev/rfd0b /dev/rfd0c:\
                .
                .
                .
st0:\
    st:\
    /dev/rst0 /dev/rst8 /dev/rst16 /dev/rst24 /dev/nrst0 /dev/nrst8 /dev/nrst16 /dev/nrst24:\
```

The `device_allocate` *File*

The administrator may modify the `device_allocate` file to change devices from allocatable to non-allocatable, or to add new devices. Here is the default `device_allocate` file shipped with the system.

```
st0;st;;;;/usr/lib/security/st_clean
fd0;fd;;;;/usr/lib/security/fd_clean
sr0;sr;;;;/usr/lib/security/sr_clean
audio;audio;;;;/usr/lib/security/audio_clean
```

The administrator defines which devices should be allocatable during initial configuration of the Basic Security Module. You may decide to accept the default devices and their defined characteristics. Whenever you add a device to any machine after the system is up and running, you must decide whether to make the new device allocatable.

The entries for devices in the `device_allocate` file may be modified by the administrator after installation. Any device that needs to be allocated before use must be defined in the `device_allocate` file on each machine.

Currently cartridge tape drives, floppy disk drives, CD-ROM devices, and audio chips are listed in the default `device_allocate(4)` file; all are allocatable and have device clean scripts. An entry for the frame buffer and four printer entries also appear in the default `device_allocate` file.

Note – If you add a Xylogics tape drive or an Archive tape drive, they can also use the `st_clean` script supplied for SCSI devices. Other devices that you could make allocatable are modems, terminals, graphics tablets, and the like, but you need to create your own device clean scripts for such devices, and the script must fulfill object reuse requirements for that type of device.

An entry in the `device_allocate` file does not mean the device is allocatable, unless the entry specifically states the device is allocatable.

In the `device_allocate` file, represent each device by a one-line entry in the following format:

device-name ; ***device-type*** ; [reserved] ; [reserved] ; [reserved] ; ***device-clean***

For example, the following line shows the entry for device name `st0`:

```
st0;st;;;;/usr/lib/security/st_clean
```

Lines in the `device_allocate(5)` can end with a ‘\’ to continue an entry on the next line. Comments may also be included. A ‘#’ makes a comment of all further text until the next NEWLINE not immediately preceded by a ‘\’.

Leading and trailing blanks are allowed in any of the fields.

device-name

Specify the name of the device (for example, `st0`, `fd0`, `sr0`). If making a new device allocatable, you can look up the device type from the device type field in the `device_maps` file, or use the `dminfo(1M)` command. (This name is also used for the name of the DAC file for the device.)

device-type

Specify the generic device type (the name for the class of devices, such as `st`, `fd`, and `sr`) — used to group related devices. If making a new device allocatable, you can look up the device type from the device type field in the `device_maps` file, or use the `dminfo(1M)` command.

device-clean

Supply the pathname of a program to be invoked for special handling, such as cleanup and object reuse protection during the allocation process. The *device clean* program is run any time the device is acted on by `deallocate(1M)`, such as when it is forcibly deallocated with `deallocate -F`.

Device Clean Scripts

The *device clean* scripts are needed to address the security requirement that all usable data is purged from a physical device before it is reused.

By default, cartridge tape drives, floppy disk drives, CD-ROM devices, and audio devices require device clean scripts, which are provided. This section describes what the device clean scripts do.

Object Reuse

Device allocation satisfies part of the object reuse requirement, but the device clean scripts also do their part by making sure that data left on a device by one user is cleared before the device is allocatable by another user.

Device Clean Scripts for Tapes

The two supported tape devices and the `device_clean` scripts for each are shown in Table 4-4:

Table 4-4 Device Clean Scripts for the Two Supported Tape Devices

Tape Device Type	Device Clean Script
SCSI 1/4 inch tape	`st_clean(1)`
Archive 1/4 inch tape	`st_clean(1)`

These scripts use the `rewoffl` option to `mt(1)` to affect the device cleanup. If the scripts are run during system boot, they will query the device to see if it is online and has any media in it.

1/4 inch tape devices will be placed in the allocate error state to force the administrator to manually clean up the device.

During the normal system operation, when `allocate` or `deallocate` is executed in the interactive mode, the user will be prompted to remove the media from the device being deallocated. The scripts pause until the media is removed from the device.

Device Clean Scripts for Floppy Disks and CD-ROM

The device_clean scripts for the floppy disk drives and CD-ROM devices are shown in Table 4-5:

Table 4-5 Device Clean Scripts for the Floppy Disk Drive and CD-ROM Device

Disk Device Type	**Device Clean Script**
Floppy Disk	fd_clean(1)
CD-ROM	sr_clean(1)

These scripts use the eject(1) command to remove the media from the drive. If eject(1) fails, the device will be placed in the allocate error state.

Device Clean Script for Audio

The audio device is cleaned up using the audio_clean(1) program. This program performs an AUDIO_DRAIN ioctl to flush the device, and then an AUDIO_SETINFO ioctl to reset the device configuration to default. In addition, this program retrieves the audio chip registers using the AUDIOGETREG ioctl, and any registers deviating from default are reset using AUDIOSETREG ioctl.

Writing New Device Clean Scripts

If you add more allocatable devices to the system, you might need to create your own device clean scripts. The deallocate command passes a parameter to the device clean scripts as shown here, a string that contains the device name (see device_allocate(4)):

```
st_clean -[I|F|S] device_name
```

Device clean scripts must return 0 for success, >0 for failure. The option letters -I|-F|-S help the script determine its running mode.

-I is needed during system boot only. All output must go to the system console. Failure or inability to forcibly eject the media must put the device in the allocate error state.

-F is for forced cleanup and -S is for standard cleanup. These are interactive and assume that the user is there to respond to prompts. With the -F option, the script must attempt to complete the cleanup if one part of the cleanup fails.

System Administration 5

New Administration Tools with Solaris 2.1

By – Janice Winsor – SunSoft, Inc.

Note – This article is a reprint of Appendix B of *Solaris System Administrator's Guide*, an award-winning retail book available in bookstores worldwide.[1]

This article describes changes to Administration Tool and introduces two new applications that are available with the Solaris 2.1 Administration Tool: *Printer Manager* and *User Account Manager*.

The *Printer Manager* provides a graphical user interface that you can use to set up the Service Access Facility for printing, to configure print servers and print clients, and to modify and delete printers.

The *User Account Manager* provides a graphical user interface that you can use to add user accounts to the `Passwd`, `Group`, `Auto_home`, and `Group` databases; create home directories; set password security; and set up the user's mail server. You can also modify and delete user accounts with the User Account Manager.

1. Janice Winsor, *Solaris System Administrator's Guide*, 2nd printing, SunSoft Press / Ziff-Davis Press, Emeryville, CA (paperback; 288 pages; ISBN 1-56276-080-7).

Changes to the Administration Tool User Interface

The Solaris 2.1 Administration Tool has some user interface changes and provides two new tools: Printer Manager and User Account Manager. Administration Tool also has a help button that you can use to display information about each of the applications. Figure 5-1 shows the Solaris 2.1 Administration Tool window.

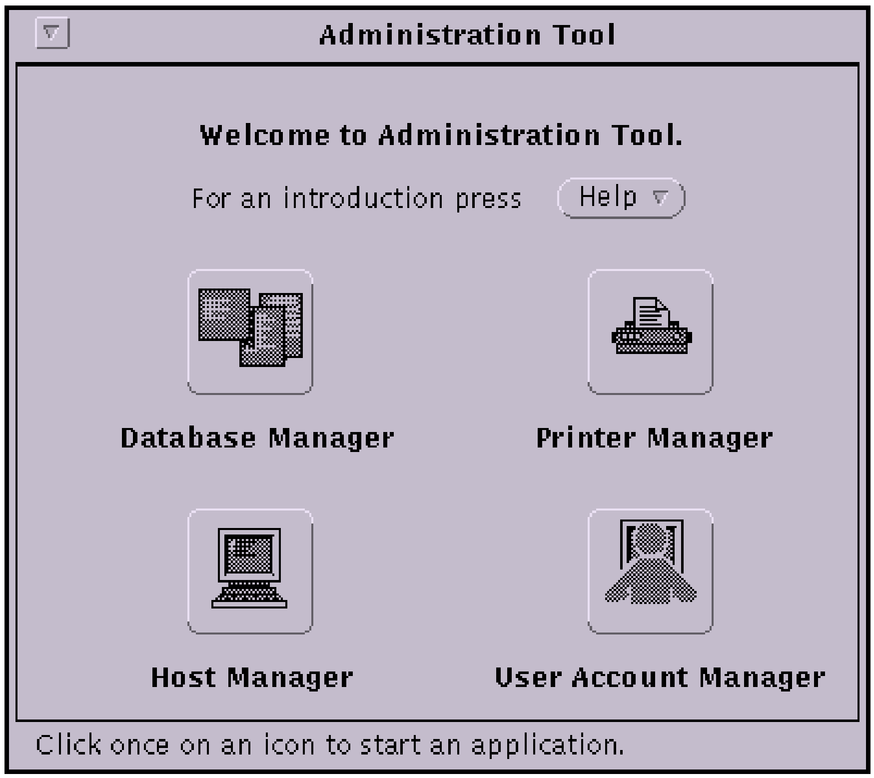


Figure 5-1 Solaris 2.1 Administration Tool window

The rest of this article introduces the Printer and User Account Manager.

Printer Manager

You can use the Printer Manager to set up Solaris 2.1 print server and print client systems. The Printer Manager automatically configures the port monitors for the Service Access Facility and updates the `LP` print service files using the information that you enter into the Printer Manager windows.

Setting Up a Print Client (Solaris 2.1)

If Solaris 2.1 is installed on the print client system, you can use the Printer Manager to set up printing. The Printer Manager automatically sets up the port monitors for Solaris 2.*x* print servers and print clients and updates the `LP` print service files.

This section describes how to use the Printer Manager to set up a print client to print on either a BSD (SunOS 4.*x*) or SVR4 (Solaris 2.*x*) print server.

To use the Administration Tool Printer Manager to set up print clients, you must be a member of the UNIX `sysadmin` group (GID `14`).

Note – The print client you want to set up must be running the Printer Manager. The Printer Manager is available with SunOS 5.1 system software. Alternatively, the print client can be a SunOS 5.0 system with the Printer Manager installed.

To set up a print client, you need this information:

- Printer name.
- Print server name.
- Type of operating system the print server is running (BSD or System V).
- Whether this is the default printer for the print client system.

Follow these steps to set up a print client:

1. If Administration Tool is not running, type `admintool &` and press `Return`. The Administration Tool window is displayed.
2. Click SELECT on the Printer Manager icon. The Printer Manager window is displayed, as shown in Figure 5-2.

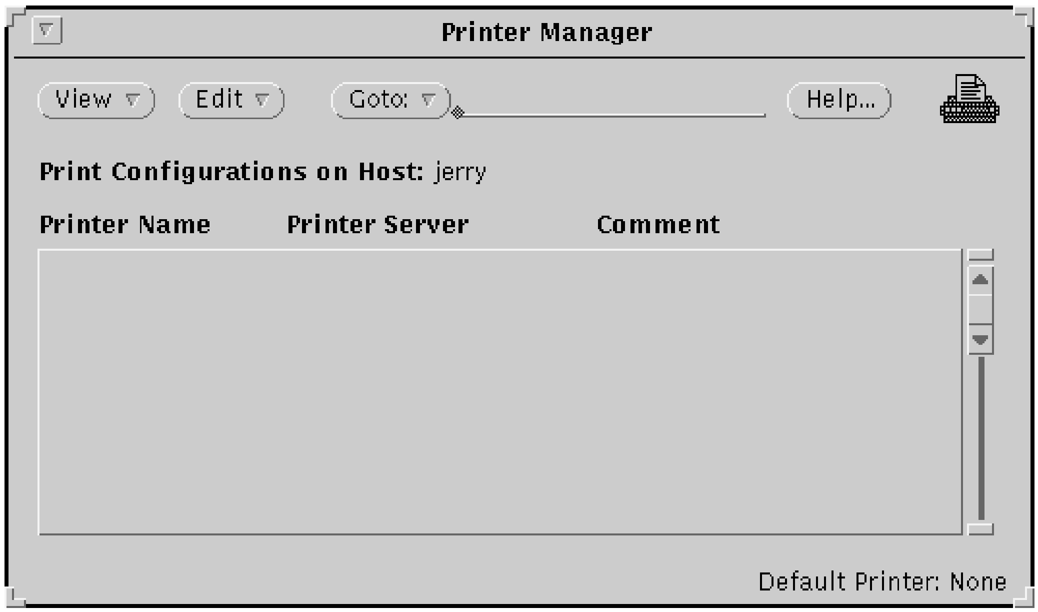


Figure 5-2 The Printer Manager window

3. If you are running the Printer Manager from the print client system, you can skip this step. If you are running the Printer Manager from a remote system, type the name of the print client system in the Goto text field and click SELECT on the Goto button.
4. Choose Add Access to Remote Printer from the Add Printer menu, as shown in Figure 5-3. The Access to Remote Printer window is displayed, as shown in Figure 5-4.

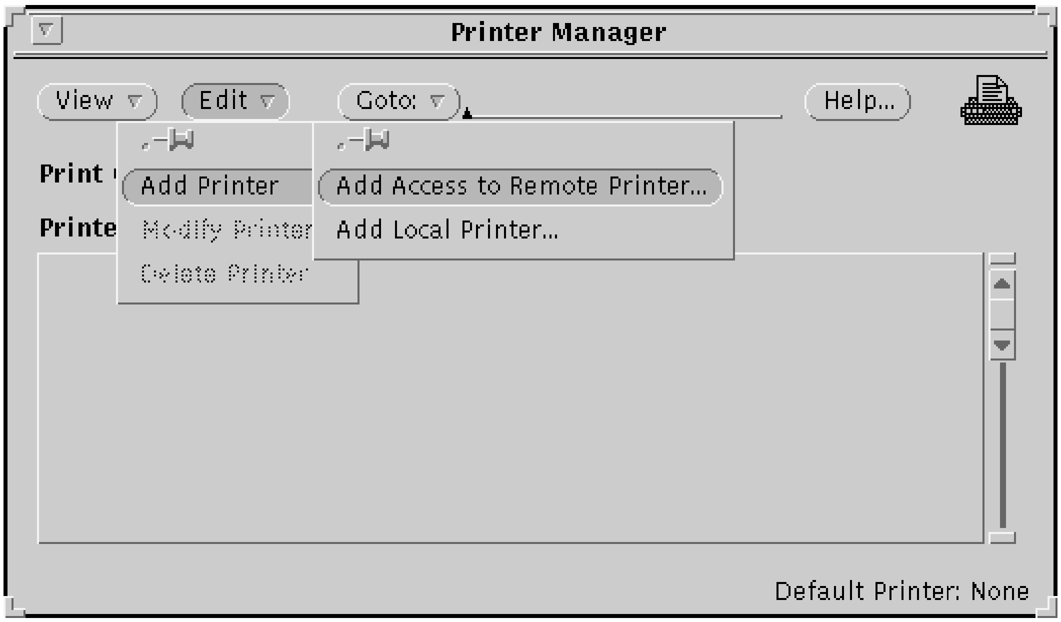


Figure 5-3 The Add Printer menu

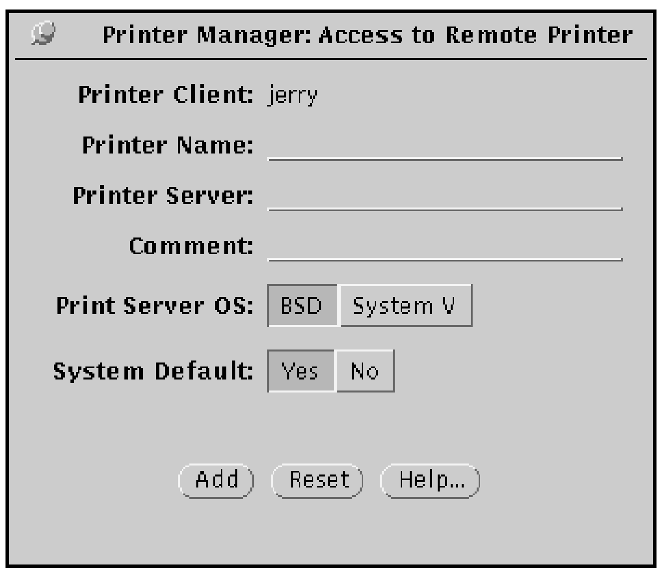


Figure 5-4 The Access to Remote Printer window

5. Type the name of the printer in the Printer Name text field.
6. Type the name of the print server in the Printer Server text field.
7. Type comments in the Comment field (optional). Include useful information such as the location of the printer or the name of the person responsible for administering it.
8. If the print server is a BSD system, go to the next step. If the print server is running SunOS 5.*x*, click SELECT on System V. You have specified the version of the operating system running on the print server.
9. The default is to use this printer as the default printer for this client system. If you do not want this printer to be the default printer, click SELECT on No.
10. Click SELECT on Add. The information defined in the Add Client window is recorded in the `/etc/lp` configuration files on the selected system, the print client, and on the specified print server system. If you make some mistakes and want to re-enter information, click SELECT on Reset to reset all the fields to their original values.

To add more clients to a print server, repeat this procedure.

The example shown in Figure 5-5 adds a remote printer named `carvel` on an SVR4 print server named `carvel`, and specifies `carvel` as the default printer for the print client system named `jerry`.

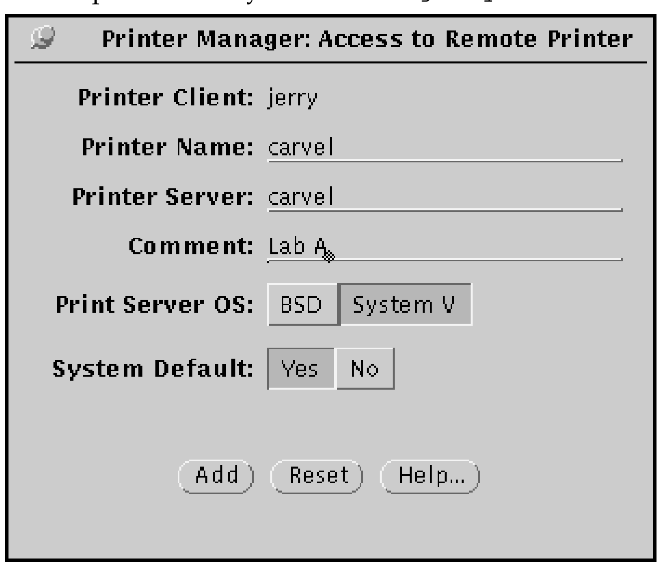


Figure 5-5 Example of adding a remote printer

Setting Up a Print Server (Solaris 2.x)

You can use the Printer Manager to automatically configure the port monitors and set up a Solaris 2.*x* print server.

To use the Administration Tool Printer Manager to set up print servers, you must be a member of the UNIX `sysadmin` group (GID `14`).

Note – The print server you want to set up must be running the Printer Manager. The Printer Manager is available with SunOS 5.1 system software. Alternatively, the print client can be a SunOS 5.0 system with the Printer Manager installed.

You need this information to set up a Solaris 2.*x* print server:

- Printer name.
- Server name.
- Device name of port to which printer is connected.
- Printer type (the default is PostScript).
- File content type (the default is PostScript).
- Fault notification policy for this print server (the default is Write to superuser).
- Whether this printer is the default printer for the print server (the default is Yes).
- Whether a banner is printed before each print job (the default is to print a banner).
- Whether you want to register this printer with the NIS+ service (the default is No). If your network is running NIS+, you can add the printer to the table of printers that are available on the network.
- Whether you want to create a specific list of users who can access this printer (the default is to permit access to all users). If you create a user list, only users who are on the list can use this printer.

Follow these steps to set up a print server:

1. If Administration Tool is not running type `admintool &` and press `Return`. The Administration Tool window is displayed.
2. Click SELECT on the Printer Manager icon. The Printer Manager window is displayed, as shown in Figure 5-6. The printers already available to the current system are listed.

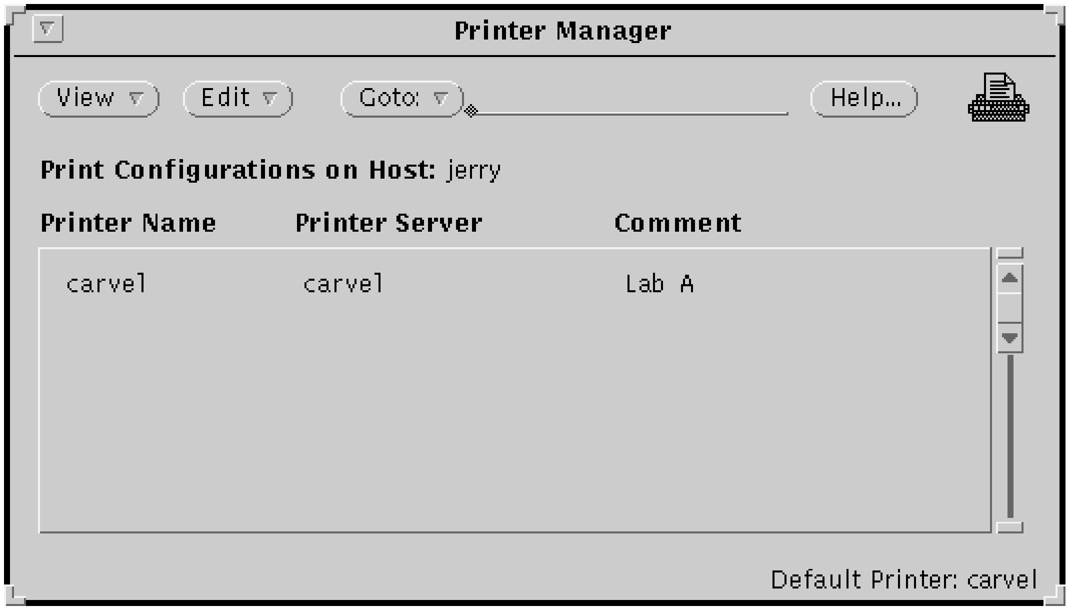


Figure 5-6 The Printer Manager window, showing current printers

3. If you are running Printer Manager from the print server, skip this step. If you are running Printer Manager from another system, type the system name in the Goto field and click SELECT on the Goto button.

4. Choose Add Printer from the Edit menu, then Add Local Printer, as shown in Figure 5-7. The Local Printer window (Figure 5-8) is displayed.

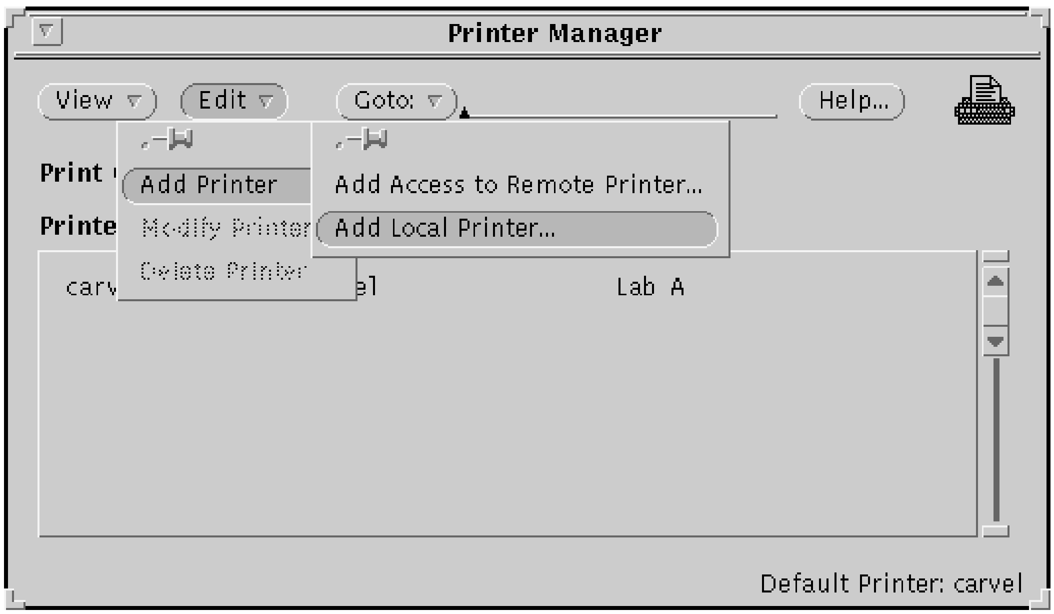


Figure 5-7 The Edit Menu

5. Type the printer name in the Printer Name text field. Choose a name unique to the network or to your administrative domain within the network.

6. Type a comment in the Comment text field. Any comment you type in this field is displayed when users request printer status.

7. Choose the appropriate printer port. If the printer port is not `/dev/term/a`, choose an item from the Printer Port menu. The Printer Manager sets the ownership and permissions on the device file to grant access to the special user `lp`.

8. Choose the printer type. If the printer type is not PostScript, choose a printer type from the Printer Type menu. A list of available printer types is constructed from the `terminfo` database. If your printer type is not included, choose Other and type the name in the pop-up window that is displayed. If you specify printer type Postscript (PS) and find that pages print in reverse order, use the Modify option under the Edit menu to change the printer type to Postscript Reverse (PSR).

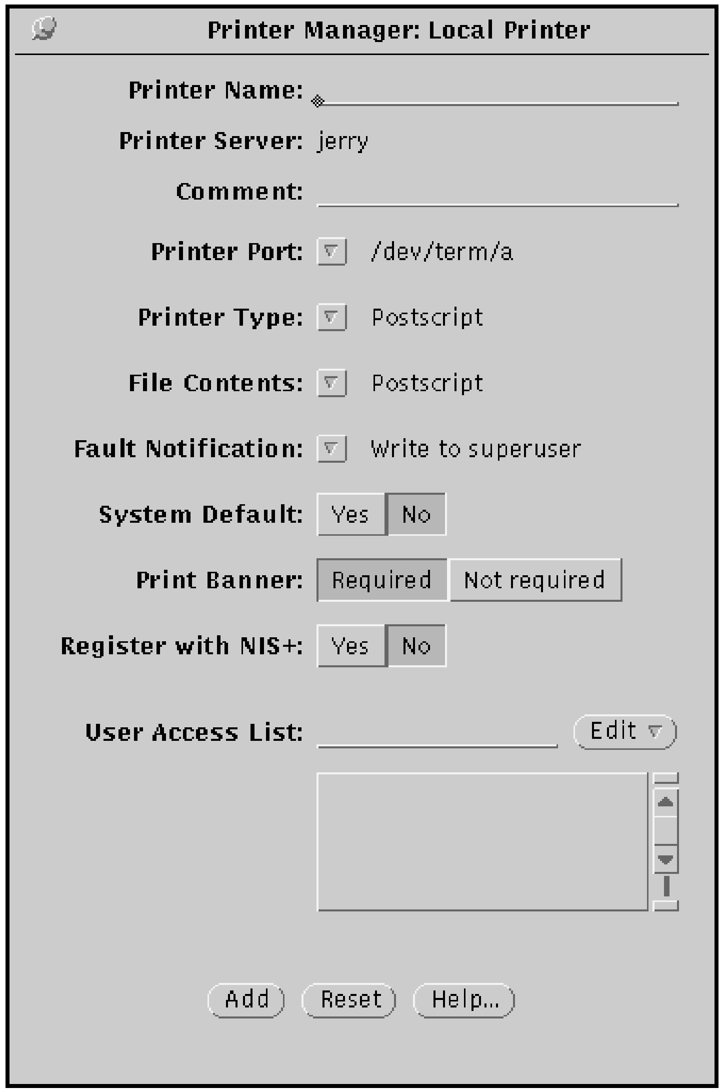


Figure 5-8 The Local Printer window

9. Choose the file contents type. If the printer is not a PostScript printer, choose a file contents type from File Contents menu. The file contents type indicates the type of file that can be printed directly on the printer without any filtering.

10. Choose the fault notification policy. If you do not want fault messages to be written to the terminal of the superuser (`root`), choose another option from the Fault Notification menu. The only other choices are to have fault messages sent by e-mail to the mailbox of the superuser (`root`), or to have no fault messages delivered at all (none).

11. Specify whether or not the printer is the default printer for this system. If you do not want the printer to be the default printer for the system to which it is being added, click SELECT on No in the System Default field.

12. Choose a banner page policy. If you do not want banner pages to be required, click SELECT on Not Required.

13. Choose whether or not to register the printer with NIS+. If you want to register the printer with NIS+, click SELECT on Yes.

14. Decide whether or not you want to restrict user access to the printer. If you want to restrict user access, type the login name of each user in the User Access List text field and click SELECT on the Edit button. Any user not in the list is denied access to the printer. If you do not create a list, all users have access to the printer.

15. Click SELECT on Add. The port monitors are configured and the information defined in the Local Printer window is recorded in the `/etc/lp` configuration files on the selected system.

User Account Manager

The User Account Manager edits the `Passwd`, `Group`, `Auto_home`, and `Cred` databases; sets password security; and creates the user's home directory. When you use the User Account Manager, you do not need to edit the appropriate databases or create the user's home directory manually. If you create more than one user with the same user profile, you can clone information from an existing user account to create a new user account.

Always run Administration Tool using your own UID, not as `root`. You must be a member of the `sysadmin` group (GID `14`). If the network is running NIS+, you also need create and delete permissions on the NIS+ databases.

Follow these steps to add a user account with the User Account Manager:

1. Click SELECT on the User Account Manager icon. The Select Naming Service window is displayed, as shown in Figure 5-9.

Figure 5-9 The Select Naming Service window

2. Choose either the NIS+ naming service or None (local `/etc` files), just as you did with the Solaris 2.0 Administration Tool. The NIS+ naming service is chosen by default and the current domain name is displayed. If you are adding a user to a different domain, type the name of the domain in the Domain Name text field. If you are adding a user to a local system, click SELECT on None and type the system name in the text field. If you select NIS, you can view information about user accounts but cannot add or modify them, because the User Account Manager does not provide methods for updating NIS maps.

3. Choose the amount of information you want displayed in the User Account Manager window using the Show menu, illustrated in Figure 5-10. By default, all users in a domain are shown in the User Account Manager window. Because it can take a while to display all of the user accounts in a domain, you can choose how much information is displayed when you open the User Account Manager.

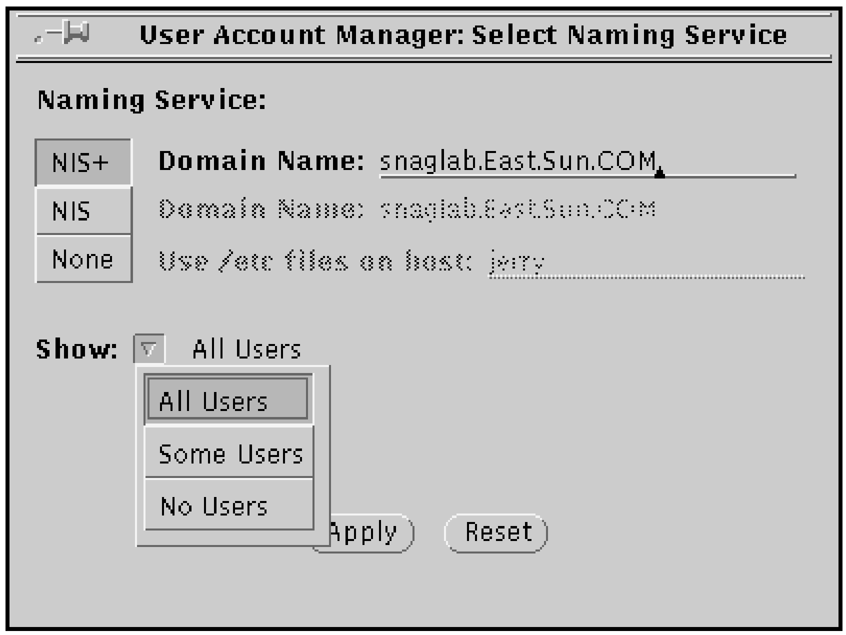


Figure 5-10 The Show menu

4. If you choose Some Users from the Show menu, a text field is displayed in the window, as shown in Figure 5-11. Type a user name or a search pattern in the text field. You can use the wildcard characters `*`, `?`, `!`, or `[]` in the text field to match entries.

Figure 5-11 The Some Users text field

5. When you have chosen the naming service and the amount of information you want displayed, click SELECT on Apply. The User Account Manager window is displayed, as shown in Figure 5-12.

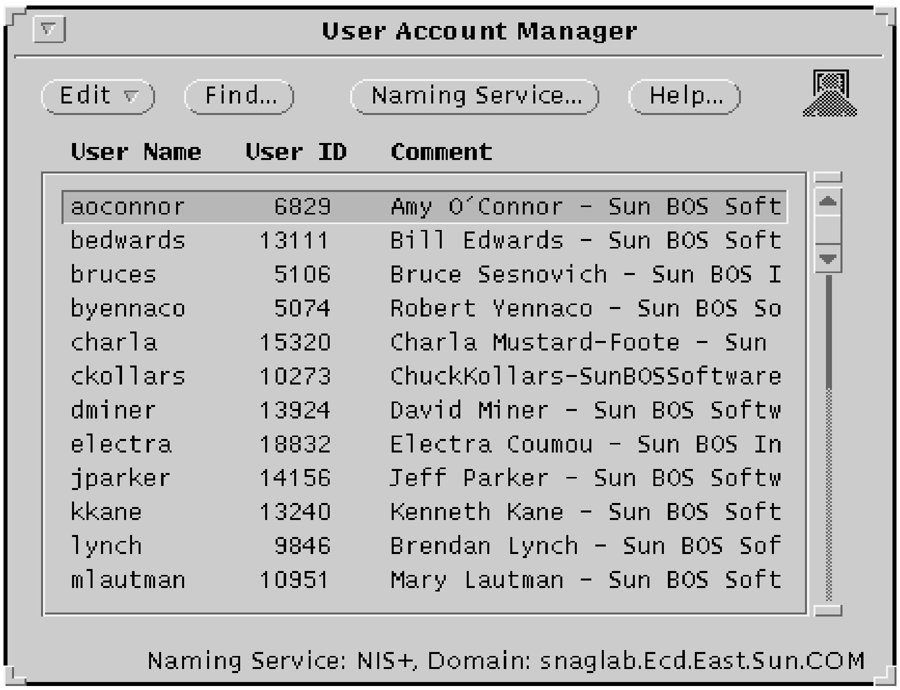


Figure 5-12 The User Account Manager window

Note – If you want to change the domain or the naming service after the User Account Manager window is opened, click SELECT on the Naming Service button. The Naming Service Window is displayed, and you can use it to change the information displayed in the User Account Manager window.

---

6. Choose Add User from the Edit menu, as shown in Figure 5-13. The Add User window is displayed, as shown in Figure 5-14. If you are copying user information, choose Copy User from the Edit menu. The Copy User window fills in all information from the existing user except the User Name and User ID text fields.

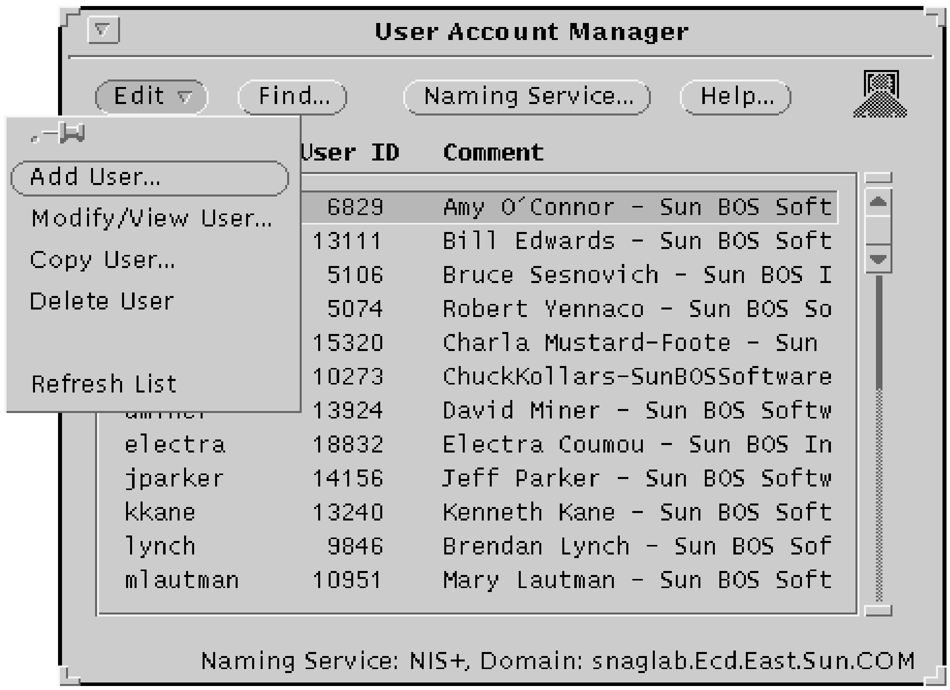


Figure 5-13 The Edit Menu

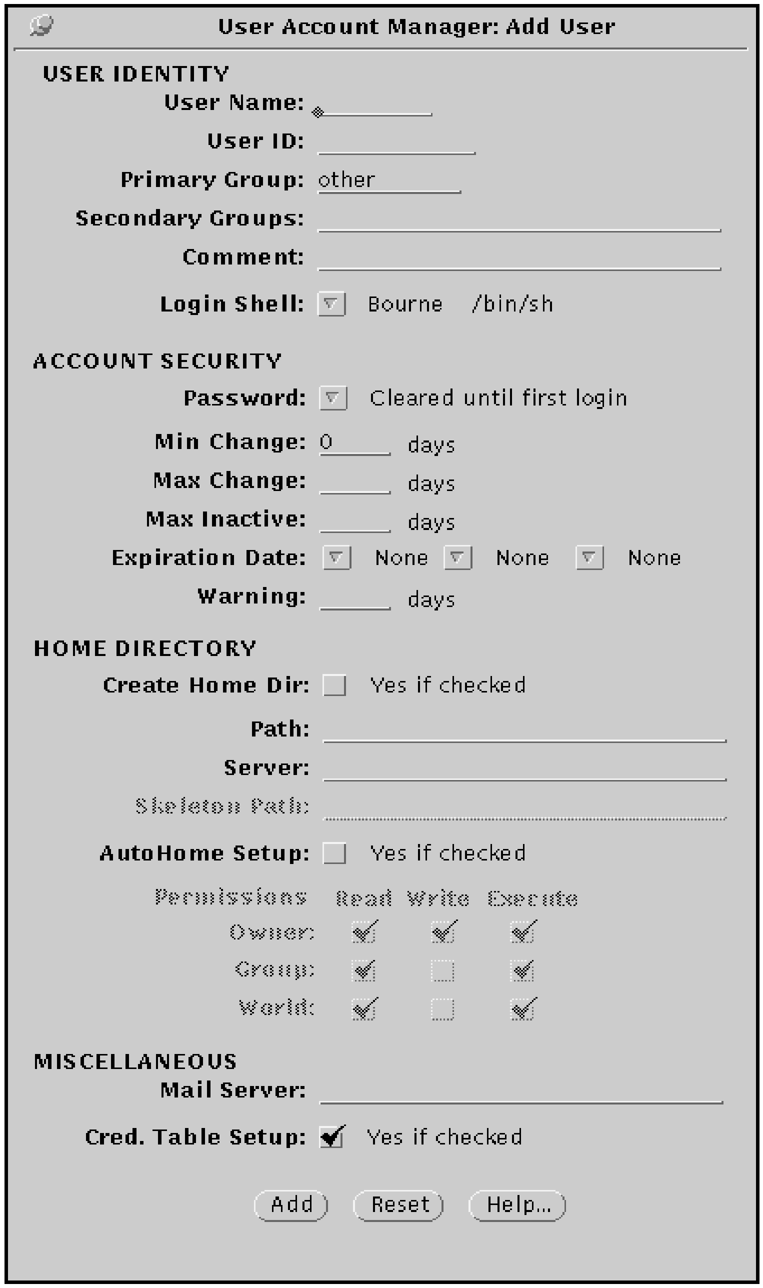


Figure 5-14 The Add User Window

7. Type the user's login name in the User Name text field. Choose a login name unique to your organization with two to eight lowercase characters and digits (excluding colons).
8. Type the UID number in the User ID text field. Choose a number between 100 and 60000 that is unique to your organization.
9. Type the user's group name or group number in the Primary Group field.
10. If the user is assigned to any secondary groups, type the names or numbers of the additional groups in the Secondary Group text field.
11. Type identifying information about the user in the Comment text field.
12. Choose a default login shell for the user from the Login Shell menu.
13. Choose a password status from the Password Status menu.
14. If you want additional password aging information, set it in the appropriate text fields.
15. If you want to automatically create the user's home directory, click SELECT on the Create Home Dir checkbox. The Skeleton Path text field is activated.
16. Type the path of the home directory to be entered in the `Passwd` database in the Path text field. If you checked the Create Home Dir box, the home directory is also created on a server.
17. Type the name of the server for the home directory in the Server text field. If you checked the Create Home Dir box, the home directory is created on a server.
18. Type the path to the directory that stores the initialization (skeleton) files that will be copied into the user's home directory in the Skeleton Path text field. Note that the initialization files and home directory must be on the same (server) system.
19. Choose whether or not you want the user's home directory to be automounted. If you choose AutoHome Setup (checkbox checked), an entry is created for the user in the `Auto_home` database. If you do not want the user's home directory to be automounted (checkbox unchecked), an entry is not created in the `Auto_home` database.
20. Click SELECT on the Permissions checkboxes to set the Read, Write, and Execute permissions for the user's home directory.

21. If you want to set up a mail account, type the name of the system that will hold the user's mail in the Mail Server text field.

22. If you want the user to be added to the NIS+ `Cred` table, make sure the Cred. Table Setup checkbox is checked.

23. When you have filled in all the information, click SELECT on Add. The information is added to the `Passwd` database, the `Group` database, the `Auto_home` database (if specified), the `Cred` database (if specified), and the `Aliases` database (if a mail server is specified). If specified, the user's home directory is created with the proper ownership and the permissions you set, and the default initialization files are copied into the user's home directory.

Modifying a User Account

To modify a user account, you must be a member of the `sysadmin` group (GID `14`). If the network is running NIS+, you also need create and delete permissions on the NIS+ databases. If you want to change a user's home directory, create the directory (`mkdir`) before making changes with the User Account Manager.

Follow these steps to modify a user account:

1. From Administration Tool, click SELECT on User Account Manager. The Select Naming Service window is displayed.

2. Choose the naming service. If you know the name(s) of the users you want to modify, choose Some Users from the Show menu and type the names of the users in the text field.

3. Click SELECT on Apply. The User Account Manager window is displayed.

4. If you want to search through the list of user names, click SELECT on Find. The Find User window is displayed, as shown in Figure 5-15.

Figure 5-15 The Find User window

5. Type the patterns you want to match in the text fields. The search looks for the first entry (in the set of entries currently displayed) that matches all of the fields. You can use wildcard characters in the search patterns.
6. Click SELECT on Find.
7. When you have found the entry you want to modify, double-click SELECT on the entry in the User Account Manager window to display the Modify User window. Alternatively, choose Modify User from the Edit menu. The Modify User window is displayed, as shown in Figure 5-16.
8. Change the information in fields, as desired.
9. Click SELECT on Apply. The user account is modified using the new information.

Figure 5-16 The Modify User window

Deleting a User Account

To delete a user account, you must be a member of the `sysadmin` group (GID `14`). If the network is running NIS+, you also need create and delete permissions on the NIS+ databases.

Follow these steps to delete a user account:

1. From Administration Tool, click SELECT on User Account Manager. The Select Naming Service window is displayed.
2. Choose the naming service. If you know the name(s) of the users you want to delete, choose Some Users from the Show menu and type the names of the users in the text field.
3. Click SELECT on Apply. The User Account Manager window is displayed.
4. Click SELECT on the entry you want to delete. The entry is highlighted.
5. Choose Delete User from the Edit menu. The Delete User window is displayed, as shown in Figure 5-17.

Figure 5-17 The Delete User window

6. If you want to delete the user's home directory and its contents, click SELECT on the appropriate checkbox.
7. If you want to delete the user's mailbox and its contents, click SELECT on the appropriate checkbox.
8. Click SELECT on Delete. The user's entry is removed from the `Passwd` database and the user is automatically removed from all groups. If specified, the home directory and its contents are deleted, as are the mailbox and its contents.

Hardware 6

Sun Type 5 Keyboard

The Sun Type 5 keyboard is a completely new design combining industry standards, innovative productivity enhancements, and high reliability. Available on all SPARCstations, the Type 5 keyboard layouts are designed to match the particular needs and preferences of both the traditional Sun user and the commercial application user.

All the Type 5 keyboards share productivity and ease-of-use features:

- Industry standard extended PC-101 layout (including numeric keypad, edit/cursor control keypad, and 12 function keys), which reduces user learning time and provides a common keyboard interface to industry software
- 11-key OPEN LOOK-style keypad
- The new "System Control Keys" keypad for operator ease-of-use
- Function LEDs located directly on keycaps (Caps Lock, Num Lock, Scroll Lock, and Compose) for improved user ergonomics and easy-to-determine key function status
- International and universal country kit versions available

The Type 5 keyboard is also available in an alternative UNIX version (North American layout only), which is the same as the standard Type 5 keyboard except for modified locations of certain keys (Caps Lock, Control, Esc[ape], and Back Space) which are in positions similar to that of the Type 4 keyboard.

Figure 6-1 illustrates the standard Type 5 keyboard in the North American layout:

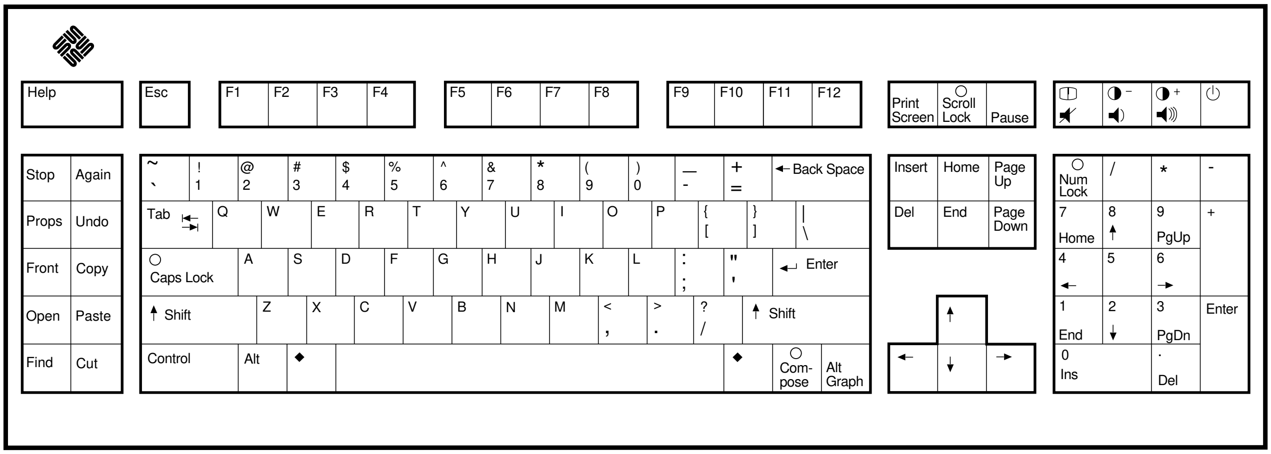


Figure 6-1 Type 5 Keyboard – North American layout

Figure 6-2 illustrates the Type 5 UNIX layout, with modified key locations indicated by shading:

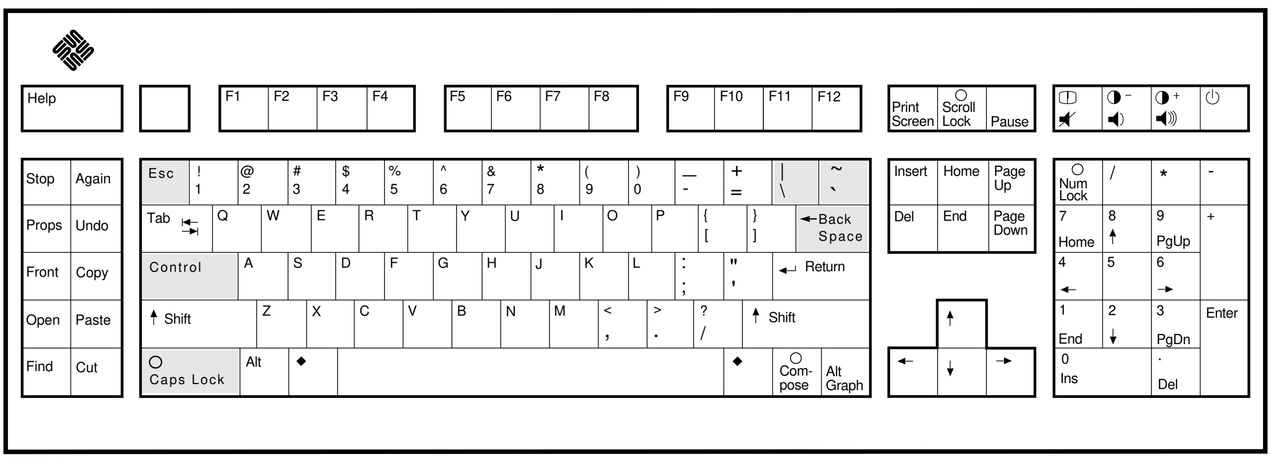
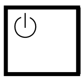


Figure 6-2 Type 5 Keyboard – UNIX layout

Power and Audio/Monitor Keys

New features of the Type 5 keyboard include the power and audio/monitor keys located in the upper-right corner.

Power Key



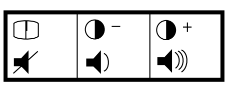
The power key, which is located in the upper-right corner of the keyboard, provides system power up and power down control.

Software and Hardware Support

The operating system documentation will contain information on the support and use of the power key when it is implemented.

The power key is not supported on the Sun-4 series, SPARCstation 1, 1+, 2, SLC, ELC, IPC, and IPX systems. If your system supports the power key, the documentation for your system will contain more information about power key support and operation.

Audio/Monitor Keys



The three audio/monitor keys are located in the upper-right corner of the keyboard. These keys provide audio volume control (mute, volume decrease, and volume increase) and monitor control (degauss, contrast decrease, and contrast increase).

Software and Hardware Support

The operating system documentation will contain information on the support and operation of these keys when it is implemented. These keys are supported on all hardware platforms that support the Type 5 keyboard.

Keycode Assignments

The keycode assignments for all Type 5 keyboards are shown below. Use these keycodes when remapping your keyboard in OpenWindows. Be sure to use the layout for your specific keyboard. See the OpenWindows documentation for more information on remapping key functions.

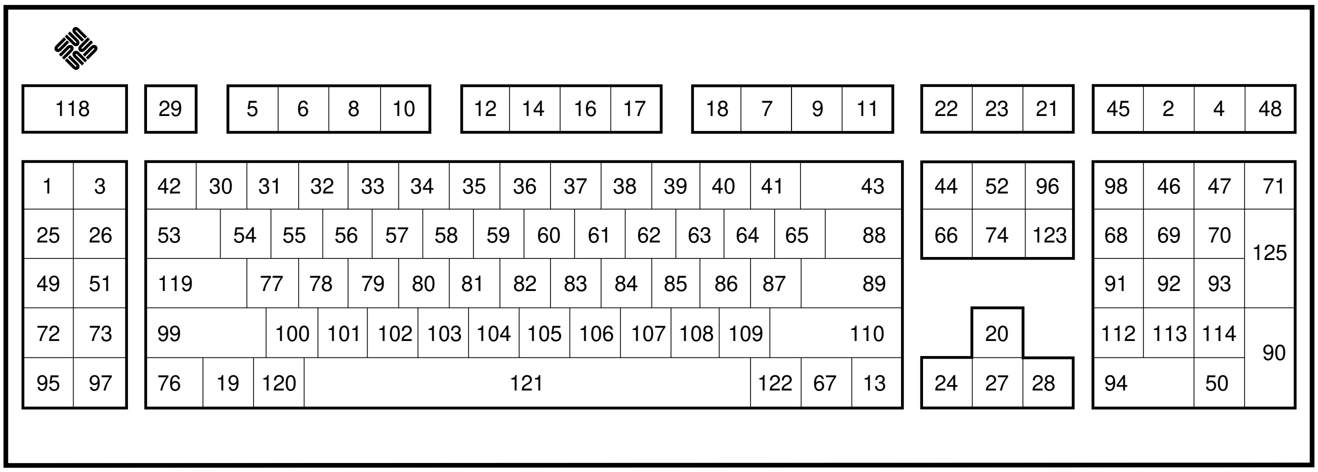


Figure 6-3 Keycode Assignments for the U.S., Taiwanese, and Korean Keyboards

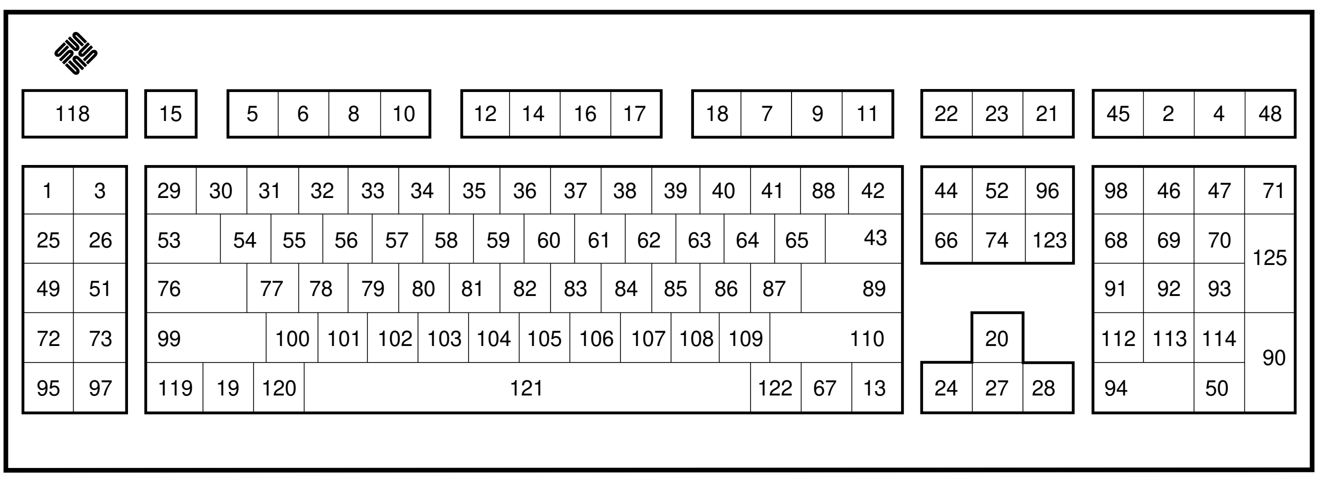


Figure 6-4 Keycode Assignments for the UNIX Keyboard

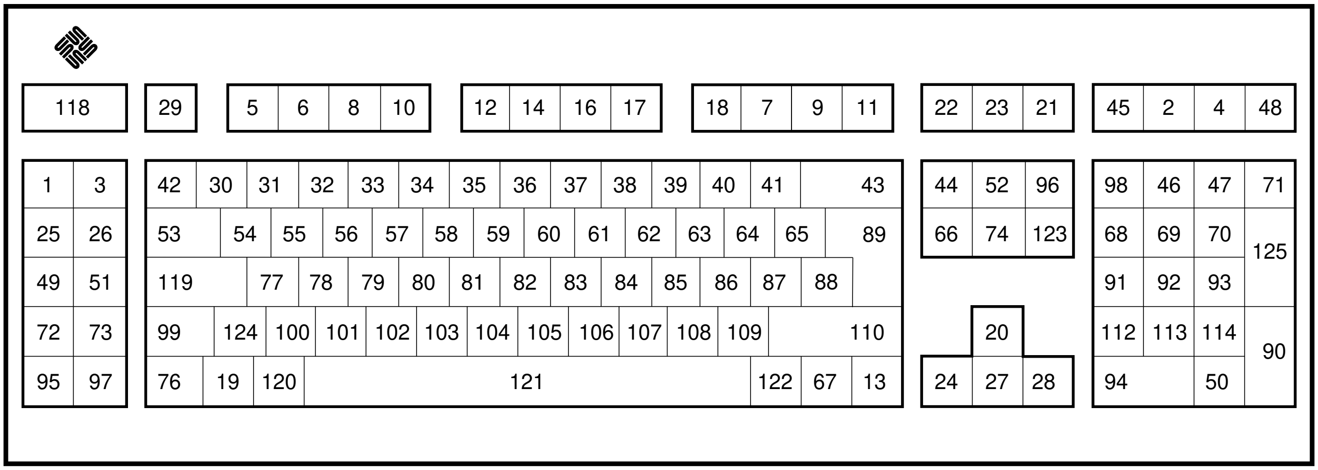


Figure 6-5 Keycode Assignments for TUV-compliant Keyboards[1]

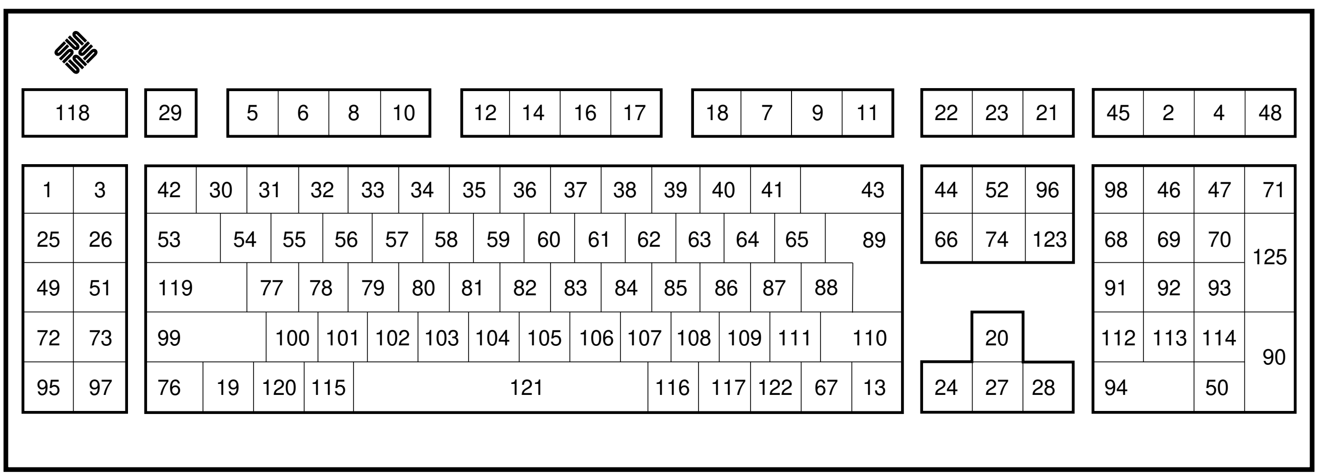


Figure 6-6 Keycode assignments for the Japanese Keyboard

1. TUV-compliant keyboards are all keyboards except for U.S., UNIX, Japanese, Korean, and Taiwanese keyboards.

R-Key Support

The Type 5 keyboard no longer supports the R-keys. On the Type 4 keyboard, these keys are located on the right keypad. However, all of the R-keys—except for the R4 key— exist on the Type 5 keyboard, but they are not marked with the R-key number and the arrangement of the keys is different than on the Type 4. To find where the R-keys are located, refer to Table 6-1 (“Comparative Mappings for the Right Keypad on Type 4 and Type 5 Keyboards”).

Remapping the Keyboard

To remap your keyboard, refer to the keymap assignments shown in Figures 6-3 through 6-6.

This example shows how to map the minus key (B15) to R4. You can map R4 to any key you want, but the minus key is a good choice because it is commonly unused and it is located near the other R-keys.

1. **Find the keymap file for your keyboard in the** `/usr/share/lib/keytables` **directory.**
   The file names for the type 5 keymaps are named by country followed by `_5`. The U.S. keyboard, for example, is named `us_5`.

2. **Add this line to the keymap file with any text editor:**

   ```
   key 71 all rf(4) num1 padminus
   ```

   This causes the minus (–) sign key to send the R4 escape sequence instead of the B15 escape sequence.

3. **Reboot your system.**
   For instructions on how to reboot your system, see the documentation for your platform.

Comparative Mappings for the Right Keypad

Table 6-1 Comparative Mappings for the Right Keypad on Type 4 and Type 5 Keyboards

Key	Type 4 Label	Type 5 Label	Comments
R1	`Pause`	`Pause`	Moved on Type 5
R2	`PrSc`	`Print Screen`	Moved on Type 5
R3	`Scroll Lock Break`	`Scroll Lock`	Moved on Type 5
R4	`=`	*none*	No equivalent key on the Type 5
R5	`/`	`/`	Same on both keyboards
R6	`*`	`*`	Same on both keyboards
R7	`Home, 7`	`Home, 7`	Same on both keyboards
R8	↑, `8`	↑, `8`	Same on both keyboards
R9	`PgUp, 9`	`PgUp, 9`	Same on both keyboards
R10	←, `4`	←, `4`	Same on both keyboards
R11	`5`	`5`	Same on both keyboards
R12	→, `6`	→, `6`	Same on both keyboards
R13	`End, 1`	`End, 1`	Same on both keyboards
R14	↓, `2`	↓, `2`	Same on both keyboards
R15	`PgDn, 3`	`PgDn, 3`	Same on both keyboards
B8	`Ins, 0`	`Ins, 0`	Same on both keyboards
B10	`Del, .`	`Del, .`	Same on both keyboards
B11	`Enter`	`Enter`	Same on both keyboards
B14	`+`	`+`	Same on both keyboards
B15	`-`	`-`	Same on both keyboards

Line Feed Key

The Type 5 keyboard no longer supports the Line Feed key. To enter a linefeed character, press Control-`j`. You can also remap it to another key.

L-Key Labels

The L-keys are no longer labeled with the L-key number on the Type 5 keyboard. They are still marked with their functional names (Stop, Again, etc.), and the functions of the keys remain the same as before.

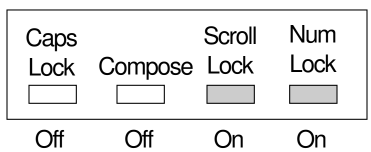
This key:	**Corresponds to:**
Stop	L1
Again	L2
Props	L3
Undo	L4
Front	L5
Copy	L6
Open	L7
Paste	L8
Find	L9
Cut	L10

Diagnostic Codes

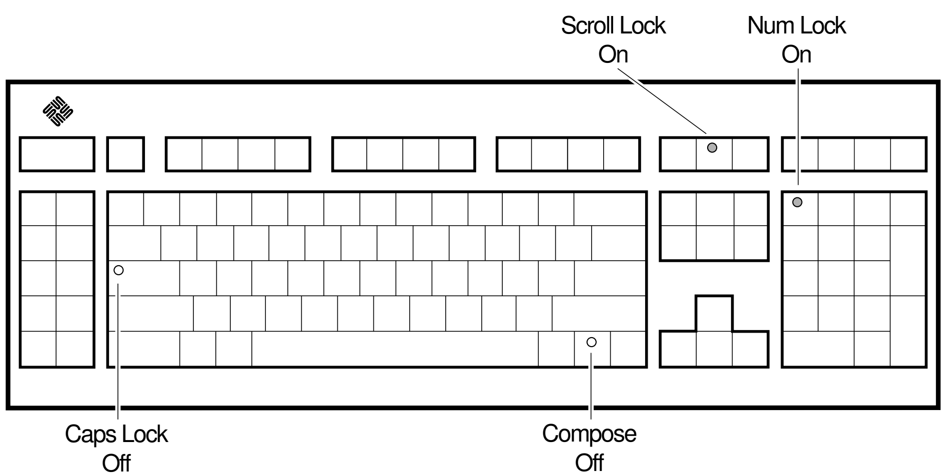
Note – This sections applies only to SPARCstation 2, ELC, and IPX systems.

On the Type 4 keyboard, diagnostic codes are displayed on the four LEDs located on the upper-right corner of the keyboard. However, on the Type 5 keyboard, the LEDs are located on the keys themselves. The diagnostic code patterns have the same meanings—only the *positions* of the LEDs have changed.

For example, a pattern of off/off/on/on on a Type 4 keyboard would appear like this:



On a U.S. Type 5 keyboard, the pattern would appear like this:



For more information on the diagnostic codes, see the documentation for your system.

Questions and Answers

Q: Will the Type 5 keyboard be compatible with my current applications?

A: Yes. The Type 5 keyboards are supported on all SPARCstations and require Solaris 1.0.1 (SunOS 4.1.2) or later. A software patch CD for SunOs 4.1 and 4.1.1 is available as an option (part number T5KBD-21). The patch CD contains software to enable the Type 5 keyboard to operate with SunOS 4.1 and 4.1.1 only. Complete instructions and software requirements are detailed in the keyboard product notes (*Sun Type 5 Keyboard Product Notes*, part number 800-6802-11). Sun-3 systems do not support the Type 5 keyboard.

OPEN LOOK applications will be fully compatible with the Type 5 keyboard. For applications not developed specifically for OPEN LOOK or which do not utilize the keyboard mapping codes included within Solaris 1.0.1 (or later releases), the Type 5 keyboard will provide comparable functionality to the Type 4 keyboard (i.e., all keys in common with Type 4 will function normally).

Questions on software compatibility, if not answered by the keyboard product notes, should be directed to the supplier of the software in question.

Q: Are the current Type 4 keyboards now obsolete or being transitioned?

A: No. Type 4 North American and universal (North American layout) country kits will continue to be offered and supported on all SPARCstations except SPARCstation 10. Type 4 European international keyboard country kits began shipping with all European systems on July 1, 1992.

The Type 4 U.S. keyboard is shown in Figure 6-7 for your reference:

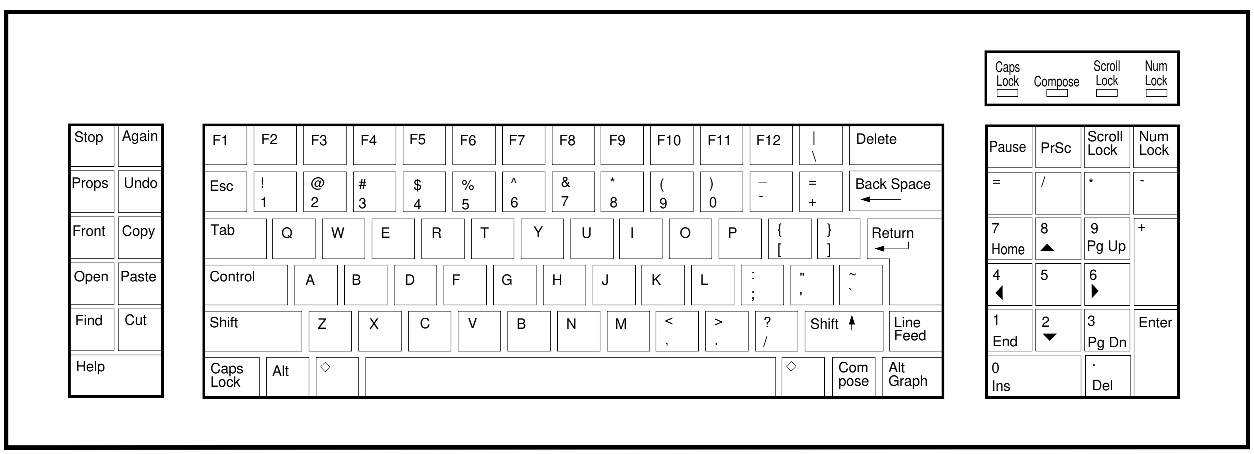


Figure 6-7 U.S. Type 4 Keyboard Layout

Q: How do I determine which keyboard to order — Type 5 or Type 5 UNIX?

A: The Sun Type 5 Standard keyboard is based on the industry standard "extended PC-101" keyboard layout. This layout is best for customers that desire a consistent and familiar keyboard interface to industry standard software applications.

The Type 5 UNIX keyboard maintains familiar key locations for Esc[ape], Control, Caps Lock, and Back Space keys (like the Type 4 keyboard). The Type 5 UNIX keyboard may be preferred by technical users and existing Sun customers familiar with our current Type 3 or Type 4 keyboards.

> **IMPORTANT:** Be sure to specify the appropriate keyboard type when placing a system order.

Q: Do all international country kits include a localized keyboard layout?

A: No. Some of the international country kits contain the universal keyboard (North American layout) due to software issues or other reasons determined by each country's marketing manager. Country kits equipped with universal keyboards provide documentation and power cords. Contact your Country Marketing Manager for further information.

Available international country kits include:

- U.S.
- UNIX
- U.K.
- French
- French-Canadian
- German
- Swiss – French
- Swiss – German
- Spanish
- Portuguese
- Italian
- Netherlands
- Swedish
- Norwegian
- Finnish
- Danish
- Japanese
- Taiwanese
- Korean

Windows 7

SunSoft's Windows Directions

SunSoft recently entered into an agreement with Adobe Systems to license the Display PostScript System (DPS) for Solaris. This announcement is intended to clarify SunSoft's OpenWindows directions in light of our partnership with Adobe and our commitment to MIT's X Window System.

Since 1989, Sun has supported MIT's X protocol in its OpenWindows product. SunSoft will continue its commitment to MIT's X Window system by shipping an OpenWindows product based on the MIT source code. The last release of Solaris featuring X11/NeWS and TNT will ship in mid-1993. Subsequent releases of Solaris will incorporate MIT's X11R5 coupled with Adobe Display PostScript Level 2, as well as other imaging and geometry libraries.

OLIT Improvements

The mid-1993 release of Solaris will feature major improvements in the OPEN LOOK Intrinsics Toolkit (OLIT) based on X11R5 intrinsics. A candidate for IEEE standardization, you can easily port applications developed with OLIT to European and Asian versions. You can dynamically customize OLIT applications at runtime, allowing better control of fonts and colors. Sporting the same look and feel of applications developed with SunSoft's popular XView toolkit, OLIT is an intrinsics-based alternative.

Why a New Direction?

Starting in 1993, OpenWindows will use MIT's source code as its foundation. Though the switch from an X11/NeWS-based server to an X11R5 server will impact applications which depend on NeWS technology, SunSoft believes developers and users will reap major benefits. Improvements in server quality and performance, as well as a more standard X implementation, will benefit all Solaris users. All programs written using the X protocol and libraries will run on the new X11R5-based server. In addition, SunSoft will be standardizing on `ddx` for the writing of graphics frame buffer device drivers. This, along with the adoption of a standard `Xinput` extension mechanism, will lower the cost of bringing hardware peripherals to market.

Questions and Answers

Q: What versions of Solaris will this new OpenWindows server run on? When will this new server ship?

A: This server will run on Solaris 2.*x* systems only.

Q: What about XView?

A: SunSoft recognizes that many of our customers use the XView toolkit. You do not need to port applications from XView to OLIT. You can maintain your present applications on XView until you feel comfortable moving to an object environment or you need functionality not available on XView.

Q: Will all the applications that now run on X11/NeWS work on this new window system? Will Drag-and-Drop and ToolTalk work?

A: If an application conforms to X protocol, it will run. As an example, Deskset (*except* PageView, which uses NeWS) runs on MIT's X11R5 with no code change. Applications using the ICCCM Drag-and-Drop protocol and ToolTalk will also work.

Unfortunately, applications using The NeWS Toolkit (TNT) or NeWS imaging will not be supported. Applications using NeWS imaging must be ported to Display PostScript. We recommend that applications using TNT be ported to OLIT.

Q: How long will X11/NeWS and TNT be supported?

A: Support for X11/NeWS and TNT will continue to be available. Source code for the latest release of OpenWindows will also be available at a reasonable price.

Q: What is DPS?

A: The Display PostScript System provides a device-independent imaging model for displaying information on the screen. By allowing you to use PostScript to display text, graphics, and sampled images, it frees you from display-specific details such as screen resolution and the number of available colors.

In September 1992, SunSoft entered into a long-term partnership with Adobe Systems to provide Solaris users with the Display PostScript System. SunSoft will feature DPS as an integral part of Solaris.

Q: Is DPS a standard?

A: Though DPS is not a sanctioned standard, it is fast becoming a de facto UNIX standard. All major UNIX platform vendors, including IBM, DEC, Silicon Graphics, etc. (but *not* Hewlett-Packard), support DPS.

Q: What do you mean by compatibility?

A: All X applications written for Solaris which conform to the X protocol should run with the X11R5 server. Developers should be able to leverage one source base for multiple UNIX platforms.

Q: What about NeWSprint?

A: NeWSprint will continue to be an integral part of Sun's printing solution. The windows direction has no adverse impact on NeWSprint. In a separate agreement with Adobe, SunPics has licensed Adobe technology for inclusion into future printing and imaging products. NeWSprint, an environment of many modules, enables customers to print on heterogenous networks. One of the modules will be Adobe's PostScript imager or RIP, which will be shared with SunSoft's Solaris OpenWindows window system. This will enrich NeWSprint by enhancing the RIP module with industry-standard Adobe PostScript.

Q: How much better is performance?

A: End users will see immediate improvements in server bootup time, application launch speed, and window snap.

Q: Is product quality that much better?

A: We believe OWV3 is a high-quality product. By moving to an MIT source base, we will be able to deliver an even higher quality product. MIT has issued many public patches since the release of R5, containing over 1300 bug fixes. Patches are issued every few weeks. Moving to the MIT X source base allows SunSoft to leverage all engineers working on X Window Systems throughout the world. SunSoft can also be more proactive in donating work to the X community at large.

Q: You say you are based on standards? Is OLIT a standard?

A: Not yet, though OLIT is now the basis of a IEEE working group — P1295.2, the Open Toolkit Environment (OTKE). OTKE is standards work which has been proposed in the IEEE Computer Society. This work will establish a toolkit standard based on OPEN LOOK.

Software Release Versions 8

Current Sun Software Products and Release Versions

The following tables contain lists of Sun software products and their respective release versions, current as of January 31, 1993. These tables appear monthly in the STB for your convenience.

Please contact your sales representative if you would like to update the release level of a Sun software product you already use, or wish to purchase another product. Use the tables to determine whether your release is the most current.

Please note the following software product change(s):

a. As of the release of Solaris 2.0, the C compiler will be unbundled.

Operating Systems

Product Name	Current Release
SunOS (Sun-4 architectures)	5.1
SunOS (Sun-3 architectures)	4.1.1_U1

Window Systems & Desktop Utilities

Product Name	Current Release
OpenWindows	3.0
SimplifySQL	1.1

Graphics Software

Product Name	Current Release
SunVision	1.2
XGL Graphics Library	2.0
SunGKS	4.1
SunPHIGS	2.0
SunPEX	1.0
Direct Xlib	1.0

Software Development

Product Name	Current Release
SPARCworks	2.0
SPARCompiler C *	2.0
SPARCompiler C++ *	3.0
SPARCompiler FORTRAN *	2.0.1
SPARCompiler Pascal *	3.0
Sun Ada Development Environment	1.1
Sun COBOL	1.0
Sun COBOL Runtime System	1.0
Sun COBOL Forms-2	1.0
NetISAM	1.0
OpenWindows Developer's Guide (DevGuide)	3.0.1
ONC / RPC Application Toolkit	1.0

* Available in both SPARCworks Professional and SPARCompiler versions

Software product availability is current for U.S. releases.
Availability may be delayed outside the U.S.

System Management

Product Name	Current Release
Online: Backup Copilot	1.0
Online: DiskSuite	1.0
Online: IPI Dual Port Tools	1.0
Sun SHIELD ARM and ASET	1.0
SunNet Manager	2.0
Sun Database Excelerator (SunDBE)	1.3

Other Products

Product Name	Current Release
NeWSprint	2.0
VideoPix	1.0
PC-NFS	4.0
PC-NFS LifeLine	2.0
PC-NFS Advanced Telnet	1.0
PC-NFS Programmer's Toolkit	4.0
SunPC	3.0

Software product availability is current for U.S. releases.
Availability may be delayed outside the U.S.

Connectivity Products

Product Name	Current Release
SunLink CG3270	7.1
SunLink SNA3270	7.0.1
SunLink BSC3270	6.1
SunLink BSCRJE	6.0
SunLink SNA Peer-to-Peer	7.0.1
SunLink Local 3270	6.1.1
SunLink Channel Gateway	7.1
SunLink DNI	7.0
SunLink TE320	7.0
SunLink OSI	7.0.1
SunNet MHS	7.0
SunLink IR	6.0
SunLink X.25	7.0
SunLink TRI/S	2.0
SunLink HSI/S	1.1
SunLink FDDI/S	1.0
Serial Parallel Controller	1.1
SunLink MCP	6.0
SunLink FDDI/DX	2.0
SunLink NetWare	1.0
SunLink Frame Relay	1.0
SunLink PPP	1.0

Software product availability is current for U.S. releases.
Availability may be delayed outside the U.S.

Updated Sun Operating System Support Matrix

When a platform has been designated as a final release, Sun's policy is to provide hardware and software support for out-of-production or transitioned systems and the latest compatible software for those systems for a minimum period of five years from the last order date.

As an ongoing policy, the level of support for SPARC products will generally decline at an increasing rate as new operating system releases are made available. Typically, the two most current versions of the operating system will be supported at level 1.

The details of Sun's software support policy have been published in a Sun Operating System Support matrix in past STB issues. As illustrated in the support matrix, older versions of the operating system generally receive lower levels of software support than newer versions. Thus, we recommend that you upgrade your system to the most current versions to ensure you receive the best quality of support from Sun.

Since we continually enhance the operating system to meet our customers' needs, we promised to keep you abreast of updates to the support matrix via the STB. Now that SunOS 4.1.3 and Solaris 2.0 are available, we provide an updated matrix.

Sun uses "support levels" to describe its software support policy. Please note that the services described in these levels are subject to local availability—contact your local service manager should you have any questions.

Level 1 Support

Level 1 offers the complete array of services contained in Sun's Personal AnswerLine Service, AnswerLine Service, and Software Update Service (see Note ①). Level 1 support also includes the full availability of options such as SunStart™, Sun's educational services, and supplementary consulting products and services.

Level 2 Support

Level 2 includes Sun's Personal AnswerLine and AnswerLine levels of phone support, limited availability of software and documentation, bug report submission and tracking, and emergency fixes and workarounds (see Note ①). Level 2 support also includes availability of supplementary consulting products and services.

Level 3 Support

Level 3 provides Time and Materials (T&M) support only.

Note ① — Software updates will only be available for the most current release. When a version of software has been designated as the final release for a platform, only critical bug fix releases will be available for that platform.

Sun Operating System Software Support Matrix

The following tables illustrate Sun's OS software support matrix. See Note ② below regarding this section.

For Sun-2	3.5	4.0/4.0.1	4.0.3	(see Note ③)
1/91-1/92	level 3	level 3	level 2	
1/92-1/94	level 3	level 3	level 3	

For Sun-3	3.5	4.0/4.0.1	4.0.3/4.0.3e	4.1	4.1.1	(see Note ③)
1/91-1/92	level 3	level 3	level 2	level 1	level 1	
1/92-1/95	level 3	level 3	level 3	level 2	level 2	
1/95-1/96	level 3	level 3	level 3	level 3	level 3	

For Sun386i	4.0.1i	4.0.2i	(see Note ③)
1/91-1/92	level 2	level 1	
1/92-1/95	level 3	level 2	
1/95-1/96	level 3	level 3	

For SPARC	3.5	4.0/4.0.1	4.0.3/4.0.3e	4.1	4.1.1	4.1.2	Solaris 2.0	4.1.3
1/91-present	level 3	level 3	level 3	level 2	level 2	level 1	level 1	level 1

Note ② — The above levels 1 and 2 are for contract customers only. Non-contract customers have the option to purchase support on a Time and Materials basis or use Sun's consulting services. The then-current unbundled products will be supported at the same level as the customer's operating system. Support for emergency fixes and workarounds may require the customer to upgrade to a more current version of the product.

Note ③ — These will be the final releases for these product lines, with the exception of critical bug fix releases. Sun's future SVR4-based operating system will function primarily on SPARC processor-based systems (Intel-based support for SVR4 is available with SunSoft's INTERACTIVE UNIX product line). Interoperability between Sun-3 or Sun386i systems and SPARC SVR4 systems will be fully supported through services such as NFS, NIS, remote login, and the like.

In addition, Sun is committed to supporting Sun-3 diskless clients running SunOS 4.1.1 on Solaris 2.0 servers.

Notes & Comments 9

World Hotlines

Sun Customers throughout the world have service hotlines available for both software and hardware support questions. The service hotlines are shown below. If your country is not shown in the table, please phone your local Sun sales office. World hotlines are divided into those for the U.S.A., Europe, and Intercon.

United States

All	**1-800-USA-4SUN** (**1-800-872-4786**)

Europe

Finland	Sun Microsystems Oy	**+358-0-5255-6300**
France	Sun Microsystems France SA	**+33-1-30-67-50-80**
Germany	Sun Microsystems GmbH	**+49-89-46008-321**
Italy	Sun Microsystems Italia SpA	**+39-6055228**
Netherlands	Sun Microsystems Nederland BV	**+33-501234**
Spain	Sun Microsystems Iberica SA	**+34-1-5560243**
Sweden	Sun Microsystems AB	**+46-8-623-9100**
Switzerland	Sun Microsystems (Schweiz) AG	**+41-1-825-7272**
	Sun Microsystems (Suisse) SA	**+41-22-364-3639**
United Kingdom		
	Sun Microsystems UK Ltd	**+44-276-691974**

Intercon

Canada:

	Quebec District Montreal, Quebec	**(514) 744-9240**, **(800) 361-1554**
	Ottawa District Ottawa, Eastern Canada	**(613) 723-8112**, **(800) 465-4415**
	Central District Toronto, Central Ontario	**(416) 490-9490**, **(800) 263-1680**
	Prairies District Manitoba, Saskatchewan, Alberta, Calgary, Western Canada	**(403) 262-6722**, **(800) 661-9256**
	Edmonton District Edmonton	**(403) 421-7264**, **(800) 661-3643**
	B. C. District Vancouver, British Columbia	**(604) 684-4120**, **(800) 663-0440**
	National CSS/Education	**(416) 477-6745**

Australia	Sun Australia	**(008) 02-4417**
New Zealand	Sun New Zealand	**(0800) 446172**
Hong Kong	Sun Asia	**852-802-4188**
Japan	Nihon Sun	**044-819-3121**

Latin America: Contact the Latin American hotline at ICON headquarters:
U.S. Office **(415) 688-9160**

Within ***Brazil***, you may also contact:
Sun do Brasil **55-11-887-9011**

Caribbean: Call your local distributor.

All Other Countries
Call **(415) 688-9372**, which is a U.S. number.

Reporting Bugs

This article contains three sections for submitting bugs. The first section describes procedures to use within the United States and Canada. The second section describes procedures within Europe. The third section describes procedures within ICON.

Submitting Software Bugs: United States and Canada

This section contains information on reporting bugs within the U.S., for customers holding and not holding support contracts.

Sun's United States Solution Center accepts software bug reports from Sun users via electronic mail and by phone. The method you use to submit a bug report varies with your needs.

U.S. users holding support contracts can report bugs to US Solution Center via the **(800) USA-4-SUN** phone hotline. Canadian users holding support contracts should call their local support center. The U.S. Solution Center phone hotline is the fastest way for a customer to find out if a problem is known and if a workaround exists. The status of previously-reported bugs can also be obtained in this way.

U.S. customers holding support contracts can also submit bug reports electronically to the address *sun!hotline* (or *hotline@Sun.COM*). Canadian customers need to use *hotline@Canada.Sun.COM*. This method generates a service order, and can be used when lines of code or other information difficult to relay over the phone is needed to describe the bug.

Customers who do not hold Sun software support contracts can report bugs via electronic mail to the address *sun!sunbugs* (or *sunbugs@Sun.COM*). These reports are reviewed periodically to determine proper disposition. Those reports determined to be from supported customers are forwarded to the U.S. Solution Center for handling. Reports from customers who cannot be verified as holding a support contract are reviewed by Sun's engineering and support personnel. An internal bug report is generated if the reported bug is new and verifiable.

Finally, customers not holding software support contracts may call the **(800) USA-4-SUN** phone hotline to report a problem and request support on a Time and Materials (T&M) basis. Canadian customers should call their local support center. In this case, please have a Purchase Order (PO) number for billing purposes.

Summary: United States and Canada

For U.S. contract customers, **(800) USA-4-SUN** is the best method to report bugs. Canadian contract customers should report bugs to their local support center. The hotlines to use appear monthly in the note entitled "World Hotlines."

The electronic mail address *sun!hotline* (or *hotline@Sun.COM*) is available for U.S. customers to submit materials that are difficult to relay over the phone. Canadian customers should use *hotline@Canada.Sun.COM* .

For non-contract customers, the electronic mail address *sun!sunbugs* (or *sunbugs@Sun.COM*) is available to report bugs. To help us serve you better, please include the following information with all electronic mail reports:

- Your name
- The name and address of your organization
- Your Sun site code, if available
- Your workstation model and serial number
- The software release(s) you are running
- A description of the problem that you are experiencing

Submitting Software Bugs: Europe

This section contains information on reporting bugs within Europe, for customers holding and not holding support contracts.

Procedures for submitting bugs are similar to those used in the United States. All customers should use their local country Answer Center to report bugs, with contract customers receiving a specific follow-up.

Sun customers not holding software service contracts can call their local Answer Center, and will need to provide a Purchase Order (PO) number at the time of the call.

Summary: Europe

To help European service centers serve you better, please include the following information with all electronic mail reports:

- Your name
- The name and address of your organization
- Your Sun site code, if available
- Your workstation model and serial number
- The software release(s) you are running
- A description of the problem that you are experiencing

Finland

Report bugs to the Finland Answer Center at the postal address shown below.

Hotline	Hotline Telephone: +358-0-5255-6300
Sun Microsystems Oy	Telefax: +358-0-5255-6252
Sinikalliontie 12	
SF-02630 Espoo, Finland	E-mail Address: *Hotline@Finland.Sun.COM*

Special Dispatch Arrangements: Provide Dispatch with the Contract number.
Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

France

Report bugs to the France Answer Center at the postal address shown below.

Service Hot Line	Hotline Telephone: +33-1-30-67-50-80
Sun Microsystems France SA	Telefax: +33-1-30-67-53-70
13, Avenue Morane Saulnier	Minitel: 36 14 code SUNFRANCE
BP 62	
78142 Velizy, France	

Special Dispatch Arrangements: Please provide Dispatch with System serial number *or* Contract number.
Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis, and order this support at the above address.

Germany

Report bugs to the Germany Answer Center at the postal address shown below.

Hotline	Hotline Telephone: +49-89-46008-321
Sun Microsystems	Telex: 5218197 sun
Bretonischer Ring 3	Telefax: +49-89-46008-320
8011 Grasbrunn, Germany	E-mail Address:
	{*sunuk,unido*}*!sunde!hotline* or
	hotline@Germany.Sun.COM

Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

Italy

Report bugs to the Italy Answer Center at the postal address shown below.

Hotline	Telephone: +039-6055228
Sun Microsystems Italia SpA	Telex: 326423 MBOX SUNITA
Centro Direzionale Colleoni	Telefax: +039-6056764
Palazzo Andromeda (Ingresso 1)	
Via Paracelso 16	
20041 Agrate Brianza	
Milan, Italy	

Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

The Netherlands

Report bugs to The Netherlands Answer Center at the postal address shown below.

Sun Microsystems Nederland BV	Hotline Telephone: +33-501234
Computerweg 1	Telefax: +33-553058
3821 AA Amersfoort	Telex: 79297 sun nl
Netherlands	

Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

Spain

Report bugs to Spain Answer Center at the postal address shown below.

Sun Microsystems Iberica SA	Hotline Telephone: +34-1-5560243
Plaza Pablo Ruiz Picasso s/n	Telefax: +34-1-5564097
Torre Picasso Planta 25	
28020 Madrid, Spain	

Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

Sweden

Report bugs to the Sweden Answer Center at the postal address shown below.

Sun Microsystems AB
Box 51
S-164 94
Kista, Sweden

Hotline Telephone: +46-8-623-9100

E-mail Address: *hotline@Sun.se* or *hotline@Sweden.Sun.COM*

Arrangements for Non-Contract Customers:
Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

Switzerland

Report bugs to the Switzerland Answer Center at one of the postal addresses shown below.

(*German*)
Sun Microsystems (Schweiz) AG
Postfach
Eschenstrasse 8
CH-8603 Schwerzenbach, Switzerland

Telephone: +41-1-825-7272
Telefax: +41-1-825-7222
E-mail: *hotline@Swiss.Sun.COM*

(*French*)
Sun Microsystems (Suisse) SA
Route des Avouillons 12
CH-1196 Gland, Switzerland

Telephone: +41-22-364-3639
Telefax: +41-22-364-1176
E-mail: *answerline@Swiss.Sun.COM*

Special Dispatch Arrangements: Provide Dispatch with the Contract number and System serial number.
Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

United Kingdom

Report bugs to the UK Answer Center at the postal address shown below.

Sun Microsystems UK Ltd
Watchmoor Park
Riverside Way
Camberley, Surrey
GU15 3YL, England

Hotline Telephone: 0276-691974
Telefax: 0276-681546

Special Dispatch Arrangements: Please provide Dispatch with either the System serial number *or* Contract number.
Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

Submitting Software Bugs: Intercon

This section contains information on reporting bugs within Intercon, for customers holding and not holding support contracts.

Procedures for submitting bugs are similar to those used in the United States. All customers should use their local country Answer Center to report bugs, with contract customers receiving a specific follow-up.

Sun customers not holding software service contracts can call their local Answer Center, and will need to provide a Purchase Order (PO) number at the time of the call.

Summary: Intercon

To help Intercon service centers serve you better, please include the following information with all electronic mail reports:

- Your name
- The name and address of your organization
- Your Sun site code, if available
- Your workstation model and serial number
- The software release(s) you are running
- A description of the problem that you are experiencing

Detailed information for individual countries follows.

Australia

Report bugs to the Australian Answer Center at the postal address shown below.

Hotline
Sun Microsystems Australia Pty. Ltd.
PO Box 320
Artarmon, New South Wales 2064
Australia

Hotline Telephone: (008) 02-4417
Telefax: 02 436 1084
E-mail Address: *hotline@Aus.Sun.COM*

Special Dispatch Arrangements: Please provide Dispatch with either the System serial number *or* Contract number.
Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

Japan

Report bugs to the Nihon Sun Answer Center at the postal address shown below.

Nihon Sun Microsystems K.K	Hotline Telephone:
KSP Technical Center - Answer Center	044-819-3121, 044-819-3111
KSP R&D Bldg. B-1F-112	FAX: 044-819-3120
Sakato 100-1, Takatsu-ku,	
Kawasaki, Kanagawa, 213	
Japan	

Special Dispatch Arrangements: Provide Dispatch with the Contract number and Serial Number.

Arrangements for Non-Contract Customers: Please provide a valid PO number for billing on a Time and Materials (T&M) basis.

New Zealand

Report bugs to the New Zealand Answer Center at the number shown below.

Hotline Telephone: (0800) 446172

Your STB Subscription

Your company's software service contract includes a monthly issue of the STB. Each month, the copy of your STB is mailed to your company's primary contact person or department named in the contract. Sites with more than one contract may receive more than one STB copy, depending on how the contracts are arranged.

Duplicating Your STB

Your primary contact person or department may duplicate this "master" STB copy for all Sun workstation end-users. As long as you duplicate copies and route them internally only, there are no copyright infringement problems.

This limited permission for duplication is for your convenience only, however, and does not include any duplication for resale, for distribution outside your company, or for distribution to employees of companies not having a Sun software service contract.

Multiple STB Subscriptions

You can now order extra copies of the STB for a low monthly fee for each additional copy. These extra copies will sent to the same address as your "master" STB copy. Please contact your local Sales/Service Office or Sun Authorized Distributor.

Changes or Updates

Changes of address or other updates, including change of contact name, should be submitted to your Customer Service Administrator (CSA), not to the STB Editor. Your CSA will register the requested changes through proper channels with Sun headquarters.

Further STB Questions

If you have any questions or comments regarding the STB, including corrections, please send your comments to *stb-editor@Sun.COM* or *sun!stb-editor*.

Index

A

Administration Tool, 5-1 to 5-22
  Printer Manager, 5-1, 5-3 to 5-11
  User Account Manager, 5-1, 5-11 to 5-22

B

Basic Security Module (BSM), 4-1 to 4-26
  `allocate`, 4-20, 4-21, 4-25
  audit daemon (`auditd`), 4-2, 4-7, 4-10 to 4-14, 4-15
  `audit_control` file, 4-4, 4-6, 4-7 to 4-8, 4-10, 4-11, 4-12, 4-13, 4-14
  `audit_user` file, 4-4, 4-6, 4-8 to 4-9, 4-10
  `audit_warn`, 4-8, 4-11 to 4-12, 4-13 to 4-14
  auditing, 4-1 to 4-18
  `auditreduce`, 4-14 to 4-17, 4-18
  `deallocate`, 4-20, 4-21, 4-24, 4-25, 4-26
  device allocation, 4-19 to 4-26
  `device_allocate` file, 4-20, 4-23 to 4-24, 4-26
  `device_maps` file, 4-20, 4-21, 4-22 to 4-23, 4-24
  `dminfo`, 4-20, 4-22, 4-24
  `list_devices`, 4-20, 4-21
  `praudit`, 4-4, 4-14, 4-15 to 4-16, 4-17 to 4-18

C

C2, 4-1, 4-19

D

Display PostScript System (DPS), 7-1, 7-3
Distributed Object Management Facility (DOMF), 2-16
Distributed Objects Everywhere (DOE), 2-2, 2-15 to 2-16

M

`mmap`, 3-2

N

NeWS, 7-1 to 7-3
NeWSprint, 7-3
NIS (Network Information Service), 4-20, 5-12
NIS+ (Network Information Service Plus), 5-7, 5-11, 5-12, 5-19, 5-22

O

ONC+, 2-5
OPEN LOOK, 6-1, 6-10
OPEN LOOK Intrinsics Toolkit (OLIT), 7-1 to 7-2, 7-4
Open Toolkit Environment (OTKE), 7-4
OpenWindows, 2-2, 2-7, 6-4, 7-1 to 7-4

P

PageView, 7-2
port monitors, 5-3, 5-7, 5-11
PostScript, 5-7, 5-9, 5-10

S

Service Access Facility (SAF), 5-3
Solaris, 2-2, 3-1, 3-2, 3-5 to 3-6, 4-1, 5-1 to 5-22, 6-10, 7-1 to 7-3
SPARCstation, 6-1, 6-3, 6-9, 6-10
swapfs, 3-2, 3-5
System V Release 4 (SVR4), 5-3, 5-6

T

The NeWS Toolkit (TNT), 7-1, 7-2 to 7-3
ToolTalk, 2-1 to 2-17, 7-2
    CASE Interoperability Message Sets, 2-14
    Data Access and Manipulation Message Set, 2-13
    Desktop Services Message Set, 2-13
    Document and Media Exchange Message Set, 2-13
Transport-Independent Remote Procedure Call (TI-RPC), 2-5
Type 5 keyboard, 6-1 to 6-11

X

X Window System, 7-1, 7-4
X11, 7-1 to 7-3
XView, 7-1, 7-2