

**Chapter
2**

**UNIX
OPERATING
SYSTEM**

*Get on the
Fast Track!*



TM

**SYS-ED/
COMPUTER
EDUCATION
TECHNIQUES, INC.**

Objectives

You will learn:

- C Process management in Solaris.
- C Components of the operating system.
- C Flow of control within the operating system.

1 Features and Facilities

The major features/facilities of the UNIX operating system include the following:

multi-user	More than one user can use the machine at a time supported remote via terminals.
multi-tasking	More than one program can be run at a time.
Hierarchical directory structure	In order to support the organization and maintenance of files.
portability	Only the kernel is written in Assembler Language. This means that the operating system can be converted easily to run on different hardware.
Program development tools	A wide range of support tools: debuggers and compilers.

2 Operating System Structure

This UNIX operating system is structured in terms of three major components.

kernel	<ul style="list-style-type: none"> C Schedules programs. C Manages data/file access and storage. C Enforces security mechanisms. C Performs all hardware access.
shell	<ul style="list-style-type: none"> C Presents each user with a prompt. C Interprets command types by a user. C Executes user commands. C Supports a custom environment for each user.
utilities	<ul style="list-style-type: none"> C File management: rm, cat, ls, rmdir, mkdir C User management: passwd, chmod, chgrp C Process management: kill, ps C Printing: lp, troff, pr C Program development tools

3 Multi-User Operating Systems

A multi-user operating system allows more than one user to share the same computer system at the same time. This is achieved by time-slicing the computer processor at regular intervals between the various users.

The switching between user programs is done in part of the kernel.

Switching from one program to another requires:

- C a regular timed interrupt event.
- C saving the interrupted programs state and data.
- C restoring the next programs state and data.
- C running that program till the next timed interrupt occurs

The timed event typically is about 1 to 10 milliseconds apart. It is generated by a real-time clock.

4 Handling Programs

Each computer has a maximum amount of memory installed. Some of this memory is required by the operating system; the remainder is available to user programs.

The more memory that can be provided the better. When there is insufficient main memory to run a user's program, some of the other users program residing in main memory must be written out to the disk in order to create free memory space. This process is called swapping.

When the system becomes overloaded and there are more users than the system can handle, the operating system spends most of its time shuttling users programs between main memory and the disk unit, and users response time degrades. This is known as disk thrashing and can be adjusted by installing more main memory.

5 Processes

Each program running on a UNIX system is called a process. When a user types a command, UNIX constructs a Process Control Block for that process.

Each process has a PCB that holds its priority, the process state, register information and additional details.

UNIX provides three utilities for managing processes.

ps	List processes.
kill	Kill a process.
&	Run a process in the background.

If the system administrator discovers that a particular user is performing an operation that has been consuming too much computing time or dominating a system resource such as a printer, the ps command can be used for identifying the offending users process followed by the kill command in order to terminate that process.

Each program is assigned a priority level. Higher priority tasks such as reading and writing to the disk are performed more regularly. User programs can have their priority adjusted dynamically, upwards or downwards, depending upon their activity and available system resources.

6 Foreground and Background Tasks

Multi-tasking systems support foreground and background tasks.

Foreground task	Has the user interacting directly with an application or operating system using the keyboard and screen.
Background task	A background task runs in the background and does not have access to the screen or keyboard. Background tasks are usually used for printing.

Running programs in the background lets the user carry on with more important tasks such as printing and formatting documents.

Background jobs can be deleted using the kill command.

7 System Start Up

When the UNIX system starts up, it also starts a system process, known as `getty`, which monitors the state of each terminal input line. When `getty` detects a user has turned their terminal on, it presents the logon prompt and once the password is validated, the UNIX system associates the shell program, `/bin/sh`, with that terminal.

Each user is presented with a shell. This is a program which displays the users prompt, handles user input and displays output on the terminal. The shell program provides a mechanism for customizing each users setup requirements, and storing this information for re-use (in the file `.profile`).

The user interacts with `/bin/sh`, which interprets each command typed.

- C Internal commands are handled within the shell - `set`, `unset`.
- C External commands are invoked as programs - `ls`, `grep`, `sort`, `ps`.

There are a number of different command line shells/user interfaces.

- C Bourne - `sh`
- C Korn - `krn`
- C C shell - `csch`
- C Bash, which is an improved Bourne shell with history and aliases.

The shell is often referred to as a command line shell, since it presents a single prompt for the user.

The user types a command, the shell invokes that command, then presents the prompt again when the command has finished. This is done on a line by line basis.

Recent enhancements have transformed the command line interface into a graphical interface (e.g. X, MOTIF, OPENVIEW) where programs are represented as objects or icons on a screen. Icons are selected and executed by utilizing a mouse.