Table 2.1 Sample Program Execution Attributes

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 M	100 M	75 M
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

Table 2.2 Effects of Multiprogramming on Resource Utilization

	Uniprogramming	Multiprogramming
Processor use	20%	40%
Memory use	33%	67%
Disk use	33%	67%
Printer use	33%	67%
Elapsed time	30 min	15 min
Throughput	6 jobs/hr	12 jobs/hr
Mean response time	18 min	10 min

Table 2.3 Batch Multiprogramming versus Time Sharing

	Batch Multiprogramming	Time Sharing
Principal objective	Maximize processor use	Minimize response time
Source of directives to operating system	Job control language commands provided with the job	Commands entered at the terminal

Table 2.4 Operating System Design Hierarchy

Level	Name	Objects	Example Operations
13	Shell	User programming environment	Statements in shell language
12	User processes	User processes	Quit, kill, suspend, resume
11	Directories	Directories	Create, destroy, attach, detach, search, list
10	Devices	External devices, such as printers, displays, and keyboards	Open, close, read, write
9	File system	Files	Create, destroy, open, close, read, write
8	Communications	Pipes	Create, destroy, open, close, read, write
7	Virtual memory	Segments, pages	Read, write, fetch
6	Local secondary store	Blocks of data, device channels	Read, write, allocate, free
5	Primitive processes	Primitive processes, semaphores, ready list	Suspend, resume, wait, signal
4	Interrupts	Interrupt-handling programs	Invoke, mask, unmask, retry
3	Procedures	Procedures, call stack, display	Mark stack, call, return
2	Instruction set	Evaluation stack, microprogram interpreter, scalar and array data	Load, store, add, subtract, branch
1	Electronic circuits	Registers, gates, buses, etc.	Clear, transfer, activate, complement

Grey shaded area represents hardware.

Table 2.5 Windows Microkernel Control Objects [MS96]

Asynchronous Procedure Call	Used to break into the execution of a specified thread and to cause a procedure to be called in a specified processor mode.
Interrupt	Used to connect an interrupt source to an interrupt service routine by means of an entry in an Interrupt Dispatch Table (IDT). Each processor has an IDT that is used to dispatch interrupts that occur on that processor.
Process	Represents the virtual address space and control information necessary for the execution of a set of thread objects. A process contains a pointer to an address map, a list of ready threads containing thread objects, a list of threads belonging to the process, the total accumulated time for all threads executing within the process, and a base priority.
Profile	Used to measure the distribution of run time within a block of code. Both user and system code can be profiled.

Table 2.6 Some Linux Signals

SIGHUP	Terminal hangup	SIGCONT	Continue
SIGQUIT	Keyboard quit	SIGTSTP	Keyboard stop
SIGTRAP	Trace trap	SIGTTOU	Terminal write
SIGBUS	Bus error	SIGXCPU	CPU limit exceeded
SIGKILL	Kill signal	SIGVTALRM	Virtual alarm clock
SIGSEGV	Segmentation violation	SIGWINCH	Window size unchanged
SIGPIPT	Broken pipe	SIGPWR	Power failure
SIGTERM	Termination	SIGRTMIN	First real-time signal
SIGCHLD	Child status unchanged	SIGRTMAX	Last real-time signal

Table 2.7 Some Linux System Calls (page 1 of 2)

Filesystem related
close Close a file descriptor.
link Make a new name for a file.
open Open and possibly create a file or device.
read Read from a file descriptor.
write Read from a file descriptor
Process related
execve Execute program.
exit Terminate the calling process.
getpid Get process identification.
setuid Set user identity of the current process.
prtrace Provides a means by which a parent process my observe and control the execution of another process, and examine and change its core image and registers.
Scheduling related
sched_getparam Sets the scheduling parameters associated with the scheduling policy for the process identified by `pid`.
sched_get_priority_max Returns the maximum priority value that can be used with the scheduling algorithm identified by `policy`.
sched_setscheduler Sets both the scheduling policy (e.g., FIFO) and the associated parameters for the process `pid`.
sched_rr_get_interval Writes into the timespec structure pointed to by the parameter `tp` the round robin time quantum for the process `pid`.
sched_yield A process can relinquish the processor voluntarily without blocking via this system call. The process will then be moved to the end of the queue for its static priority and a new process gets to run.

Table 2.7 Some Linux System Calls (page 2 of 2)

Interprocess Communication (IPC) related
msgrcv A message buffer structure is allocated to receive a message. The system call then reads a message from the message queue specified by msqid into the newly created message buffer.
semctl Performs the control operation specified by `cmd` on the semaphore set `semid`.
semop Performs operations on selected members of the semaphore set `semid`.
shmat Attaches the shared memory segment identified by `shmid` to the data segment of the calling process.
shmctl Allows the user to receive information on a shared memory segment, set the owner, group, and permissions of a shared memory segment, or destroy a segment.
Socket (networking) related
bind Assigns the local IP address and port for a socket. Returns 0 for success and –1 for error.
connect Establishes a connection between the given socket and the remote socket associated with sockaddr.
gethostname Returns local host name.
send Send the bytes contained in buffer pointed to by *msg over the given socket.
setsockopt Sets the options on a socket
Miscellaneous
create_module Attempts to create a loadable module entry and reserve the kernel memory that will be needed to hold the module.
fsync Copies all in-core parts of a file to disk, and waits until the device reports that all parts are on stable storage.
query_module Requests information related to loadable modules from the kernel.
time Returns the time in seconds since January 1, 1970.
vhangup Simulates a hangup on the current terminal. This call arranges for other users to have a "clean" tty at login time.