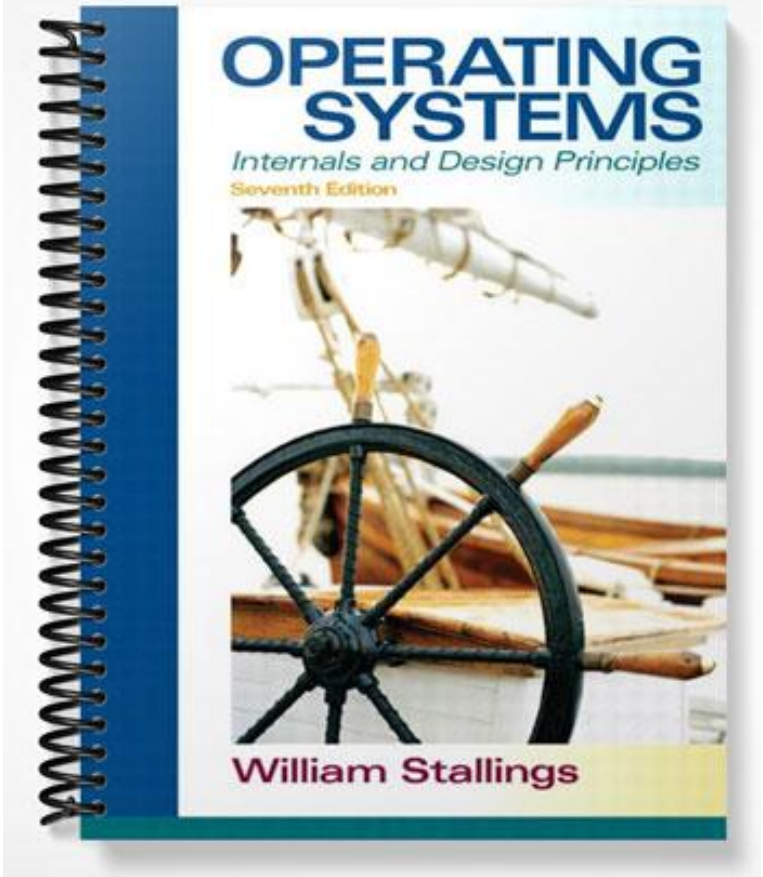
SOLUTIONS MANUAL



CHAPTER 2 OPERATING SYSTEM OVERVIEW

ANSWERS TO QUESTIONS

2.1 **Convenience:** An operating system makes a computer more convenient to use. **Efficiency:** An operating system allows the computer system resources to be used in an efficient manner. **Ability to evolve:** An operating system should be constructed in such a way as to permit the effective development, testing, and introduction of new system functions without interfering with service.

2.2 The kernel is a portion of the operating system that includes the most heavily used portions of software. Generally, the kernel is maintained permanently in main memory. The kernel runs in a privileged mode and responds to calls from processes and interrupts from devices.

2.3 Multiprogramming is a mode of operation that provides for the interleaved execution of two or more computer programs by a single processor.

2.4 A process is a program in execution. A process is controlled and scheduled by the operating system.

2.5 The **execution context,** or **process state,** is the internal data by which the operating system is able to supervise and control the process. This internal information is separated from the process, because the operating system has information not permitted to the process. The context includes all of the information that the operating system needs to manage the process and that the processor needs to execute the process properly. The context includes the contents of the various processor registers, such as the program counter and data registers. It also includes information of use to the operating system, such as the priority of the process and whether the process is waiting for the completion of a particular I/O event.

2.6 **Process isolation:** The operating system must prevent independent processes from interfering with each other's memory, both data and instructions. **Automatic allocation and management:** Programs should be dynamically allocated across the memory hierarchy as required. Allocation should be transparent to the programmer. Thus, the programmer is relieved of concerns relating to memory limitations, and the operating system can achieve efficiency by assigning memory to jobs only as needed. **Support of modular programming:** Programmers should be able to define program modules, and to create, destroy, and alter the size of modules dynamically. **Protection and access control:** Sharing of memory, at any level of the memory hierarchy, creates the potential for one program to address the memory space of another. This is desirable when sharing is needed by particular applications. At other times, it threatens the integrity of programs and even of the operating system itself. The operating system must allow portions of memory to be accessible in various ways by various users. **Long-term storage:** Many application programs require means for storing information for extended periods of time, after the computer has been powered down.

2.7 A **virtual address** refers to a memory location in virtual memory. That location is on disk and at some times in main memory. A real address is an address in main memory.

2.8 Round robin is a scheduling algorithm in which processes are activated in a fixed cyclic order; that is, all processes are in a circular queue. A process that cannot proceed because it is waiting for some event (e.g. termination of a child process or an input/output operation) returns control to the scheduler.

2.9 A **monolithic kernel** is a large kernel containing virtually the complete operating system, including scheduling, file system, device drivers, and memory management. All the functional components of the kernel have access to all of its internal data structures and routines. Typically, a monolithic kernel is implemented as a single process, with all elements sharing the same address space. A **microkernel** is a small privileged operating system core that provides process scheduling, memory management, and communication services and relies on other processes to perform some of the functions traditionally associated with the operating system kernel.

2.10 Multithreading is a technique in which a process, executing an application, is divided into threads that can run concurrently.

2.11 Simultaneous concurrent processes or threads; scheduling; synchronization; memory management; reliability and fault tolerance.

ANSWERS TO PROBLEMS

2.1 The answers are the same for **(a)** and **(b)**. Assume that although processor operations cannot overlap, I/O operations can.

Number of jobs	TAT	Throughput	Processor utilization
1	NT	1/N	50%
2	NT	2/N	100%
4	(2N – 1)T	4/(2N – 1)	100%

2.2 I/O-bound programs use relatively little processor time and are therefore favored by the algorithm. However, if a processor-bound process is denied processor time for a sufficiently long period of time, the same algorithm will grant the processor to that process since it has not used the processor at all in the recent past. Therefore, a processor-bound process will not be permanently denied access.

2.3 With time sharing, the concern is turnaround time. Time-slicing is preferred because it gives all processes access to the processor over a short period of time. In a batch system, the concern is with throughput, and the less context switching, the more processing time is available for the processes. Therefore, policies that minimize context switching are favored.

2.4 A system call is used by an application program to invoke a function provided by the operating system. Typically, the system call results in transfer to a system program that runs in kernel mode.

2.5 The system operator can review this quantity to determine the degree of "stress" on the system. By reducing the number of active jobs allowed on the system, this average can be kept high. A typical guideline is that this average should be kept above 2 minutes [IBM86]. This may seem like a lot, but it isn't.

2.6 **a.** If a conservative policy is used, at most 20/4 = 5 processes can be active simultaneously. Because one of the drives allocated to each process can be idle most of the time, at most 5 drives will be idle at a time. In the best case, none of the drives will be idle.

b. To improve drive utilization, each process can be initially allocated with three tape drives. The fourth one will be allocated on demand. In this policy, at most $\lfloor 20/3 \rfloor = 6$ processes can be active simultaneously. The minimum number of idle drives is 0 and the maximum number is 2.