

Chapter 1 – Introduction to Operating Systems

Outline

- 1.1 Introduction
- 1.2 What Is an Operating System?
- 1.3 Early History: The 1940s and 1950s
- 1.4 The 1960s
- 1.5 The 1970s
- 1.6 The 1980s
- 1.7 History of the Internet and World Wide Web
- 1.8 The 1990s
- 1.9 2000 and Beyond
- 1.10 Application Bases
- 1.11 Operating System Environments
- 1.12 Operating System Components and Goals
 - 1.12.1 Core Operating System Components
 - 1.12.2 Operating System Goals



Chapter 1 – Introduction to Operating Systems

Outline (continued)

1.13 Operating System Architectures

1.13.1 Monolithic Architecture

1.13.2 Layered Architecture

1.13.3 Microkernel Architecture

1.13.4 Networked and Distributed Operating Systems



Objectives

- After reading this chapter, you should understand:
 - what an operating system is.
 - a brief history of operating systems.
 - a brief history of the Internet and the World Wide Web.
 - core operating system components.
 - goals of operating systems.
 - operating system architectures.



1.1 Introduction

- Unprecedented growth of computing during the past several decades.
- Desktop workstations execute billions of instructions per second (BIPS)
- Supercomputers can execute over a trillion instructions per second
- Computers are now employed in almost every aspect of life.



1.2 What Is an Operating System?

- Some years ago an operating system was defined as the software that controls the hardware.
- Landscape of computer systems has evolved significantly, requiring a more complicated definition.
- Applications are now designed to execute concurrently.



1.2 What Is an Operating System?

- Separates applications from the hardware they access
 - Software layer
 - Manages software and hardware to produce desired results
- Operating systems primarily are resource managers
 - Hardware
 - Processors
 - Memory
 - Input/output devices
 - Communication devices
 - Software applications



1.3 Early History: The 1940s and 1950s

- Operating systems evolved through several phases
 - 1940s
 - Early computers did not include operating systems
 - 1950s
 - Executed one job at a time
 - Included technologies to smooth job-to-job transitions
 - Single-stream batch-processing systems
 - Programs and data submitted consecutively on tape



1.4 The 1960s

- 1960s
 - Still batch-processing systems
 - Process multiple jobs at once
 - Multiprogramming
 - One job could use processor while other jobs used peripheral devices
 - Advanced operating systems developed to service multiple interactive users
- 1964
 - IBM announced System/360 family of computers



1.4 The 1960s

- Timesharing systems
 - Developed to support many simultaneous interactive users
 - Turnaround time was reduced to minutes or seconds
 - Time between submission of job and the return of its results
 - Real-time systems
 - Supply response within certain bounded time period
 - Improved development time and methods
 - MIT used CTSS system to develop its own successor, Multics
 - TSS, Multics and CP/CMS all incorporated virtual memory
 - Address more memory locations than actually exist



1.5 The 1970s

- Primarily multimode timesharing systems
 - Supported batch processing, timesharing and real-time applications
 - Personal computing only in incipient stages
 - Fostered by early developments in microprocessor technology
- Department of Defense develops TCP/IP
 - Standard communications protocol
 - Widely used in military and university settings
 - Security problems
 - Growing volumes of information passed over vulnerable communications lines.



1.6 The 1980s

- 1980s
 - Decade of personal computers and workstations
 - Computing distributed to sites at which it was needed
 - Personal computers proved relatively easy to learn and use
 - Graphical user interfaces (GUI)
 - Transferring information between computers via networks became more economical and practical



1.6 The 1980s

- Client/server computing model became widespread
 - Clients request various services
 - Servers perform requested services
- Software engineering field continued to evolve
 - Major thrust by the United States government aimed at tighter control of Department of Defense software projects
 - Realizing code reusability
 - Greater degree of abstraction in programming languages
 - Multiple threads of instructions that could execute independently



1.7 History of the Internet and World Wide Web

- Advanced Research Projects Agency (ARPA)
 - Department of Defense
 - In late 1960s, created and implemented ARPAnet
 - Grandparent of today's Internet
 - Networked main computer systems of ARPA-funded institutions
 - Capable of near-instant communication via e-mail
 - Designed to operate without centralized control



1.7 History of the Internet and World Wide Web

- Transmission Control Protocol/Internet Protocol
 - Set of rules for communicating over ARPANet
 - TCP/IP manages communication between applications
 - Ensure that messages routed properly from sender to receiver
 - Error-correction
 - Later opened to general commercial use



1.7 History of the Internet and World Wide Web

- World Wide Web (WWW)
 - Locate and view multimedia-based documents on almost any subject
 - Early development begun in 1989 at CERN by Tim Berners-Lee
 - Technology for sharing information via hyperlinked text documents
 - HyperText Markup Language (HTML)
 - Defines documents on WWW
 - Hypertext Transfer Protocol (HTTP)
 - Communications backbone used to transfer documents across WWW



1.8 The 1990s

- Hardware performance improved exponentially
 - Inexpensive processing power and storage
 - Execute large, complex programs on personal computers.
 - Economical machines for extensive database and processing jobs
 - Mainframes rarely necessary
 - Shift toward distributed computing rapidly accelerated
 - Multiple independent computers performing common task



1.8 The 1990s

- Operating system support for networking tasks became standard
 - Increased productivity and communication
- Microsoft Corporation became dominant
 - Windows operating systems
 - Employed many concepts used in early Macintosh operating systems
 - Enabled users to navigate multiple concurrent applications with ease.
- Object technology became popular in many areas of computing
 - Many applications written in object-oriented programming languages
 - For example, C++ or Java
 - Object-oriented operating systems (OOOS)
 - Objects represent components of the operating system
 - Concepts such as inheritance and interfaces
 - Exploited to create modular operating systems
 - Easier to maintain and extend than systems built with previous techniques



1.8 The 1990s

- Most commercial software sold as object code
 - The source code not included
 - Enables vendors to hide proprietary information and programming techniques
- Free and open-source software became increasingly common in the 1990s
 - Open-source software distributed with the source code
 - Allows individuals to examine and modify software
 - Linux operating system and Apache Web server both open-source
- Richard Stallman launched the GNU project
 - Recreate and extend tools for AT&T's UNIX operating system
 - He disagreed with concept of paying for permission to use software



1.8 The 1990s

- Open Source Initiative (OSI)
 - Founded to further benefits of open-source programming
 - Facilitates enhancements to software products
 - Permits anyone to test, debug and enhance applications
 - Increases chance that subtle bugs will be caught and fixed
 - Crucial for security errors which need to be fixed quickly
 - Individuals and corporations can modify the source
 - Create custom software to meet needs of certain environment



1.8 The 1990s

- Operating systems became increasingly user friendly
 - GUI features pioneered by Apple widely used and improved
 - “Plug-and-play” capabilities built into operating systems
 - Enable users to add and remove hardware components dynamically
 - No need to manually reconfigure operating system



1.9 2000 and Beyond

- **Middleware**
 - Links two separate applications
 - Often over a network and between incompatible machines
 - Particularly important for Web services
 - Simplifies communication across multiple architectures
- **Web services**
 - Encompass set of related standards
 - Ready-to-use pieces of software on the Internet
 - Enable any two applications to communicate and exchange data



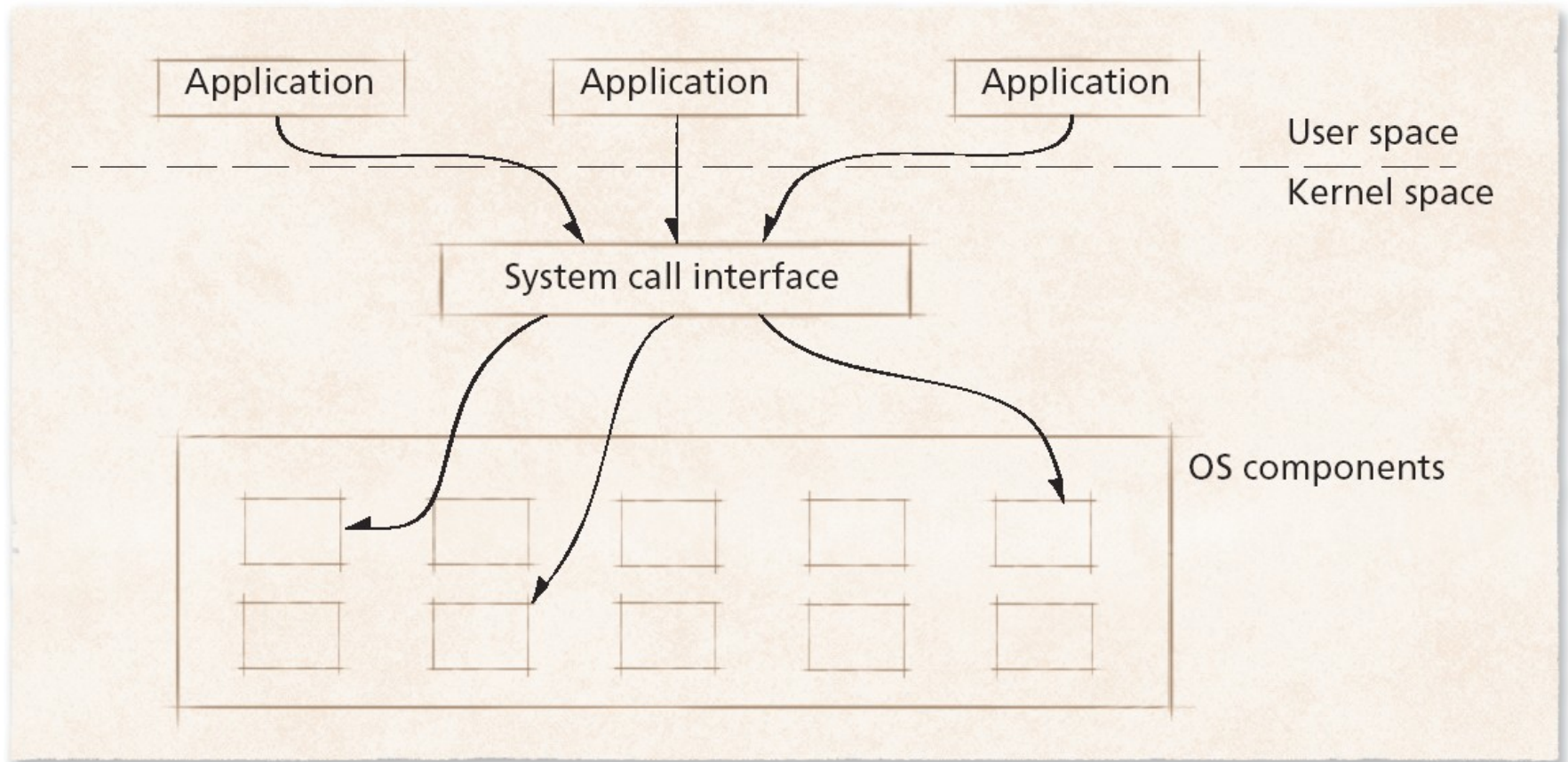
1.10 Application Bases

- IBM PC immediately spawned a huge software industry
 - Independent software vendors (ISVs) market software packages to run under MS-DOS operating system.
 - Operating system must present environment conducive to rapid and easy application development
 - Otherwise unlikely to be adopted widely
- Application base
 - Combination of hardware and operating system used to develop applications
 - Developers and users unwilling to abandon established application base
 - Increased financial cost and time spent relearning



1.10 Application Bases

Figure 1.1 Interaction between applications and the operating system.



1.11 Operating System Environments

- Operating systems intended for high-end environments
 - Special design requirements and hardware support needs
 - Large main memory
 - Special-purpose hardware
 - Large numbers of processes
- Embedded systems
 - Characterized by small set of specialized resources
 - Provide functionality to devices such as cell phones and PDAs
 - Efficient resource management key to building successful operating system



1.11 Operating System Environments

- Real-time systems
 - Require that tasks be performed within particular (often short) time frame
 - Autopilot feature of an aircraft must constantly adjust speed, altitude and direction
 - Such actions cannot wait indefinitely—and sometimes cannot wait at all



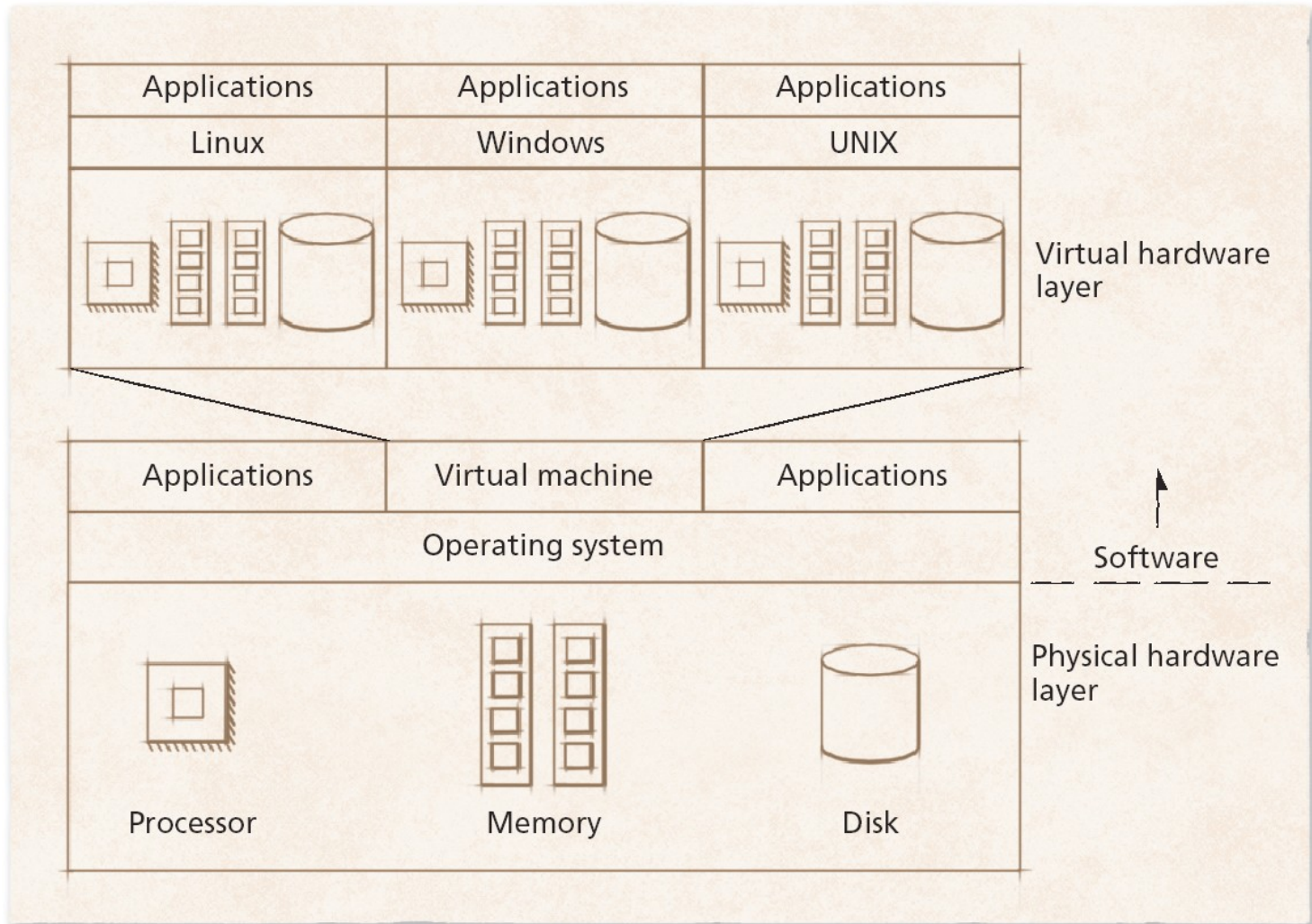
1.11 Operating System Environments

- Virtual machines (VMs)
 - Software abstraction of a computer
 - Often executes on top of native operating system
- Virtual machine operating system
 - Manages resources provided by virtual machine
- Applications of virtual machines
 - Allow multiple instances of an operating system to execute concurrently
 - Emulation
 - Software or hardware mimics functionality of hardware or software not present in system
 - Promote portability



1.11 Operating System Environments

Figure 1.2 Schematic of a virtual machine.



1.12 Operating System Components and Goals

- Computer systems have evolved
 - Early systems contained no operating system,
 - Later gained multiprogramming and timesharing machines
 - Personal computers and finally truly distributed systems
 - Filled new roles as demand changed and grew



1.12.1 Core Operating System Components

- User interaction with operating system
 - Often, through special application called a shell
 - Kernel
 - Software that contains core components of operating system
- Typical operating system components include:
 - Processor scheduler
 - Memory manager
 - I/O manager
 - Interprocess communication (IPC) manager
 - File system manager



1.12.1 Core Operating System Components

- Multiprogrammed environments now common
 - Kernel manages the execution of processes
 - Program components which execute independently but use single memory space to share data are called threads.
 - To access I/O device, process must issue system call
 - Handled by device driver
 - Software component that interacts directly with hardware
 - Often contains device-specific commands



1.12.2 Operating System Goals

- Users expect certain properties of operating systems
 - Efficiency
 - Robustness
 - Scalability
 - Extensibility
 - Portability
 - Security
 - Protection
 - Interactivity
 - Usability



1.13 Operating System Architectures

- Today's operating systems tend to be complex
 - Provide many services
 - Support variety of hardware and software
 - Operating system architectures help manage this complexity
 - Organize operating system components
 - Specify privilege with which each component executes



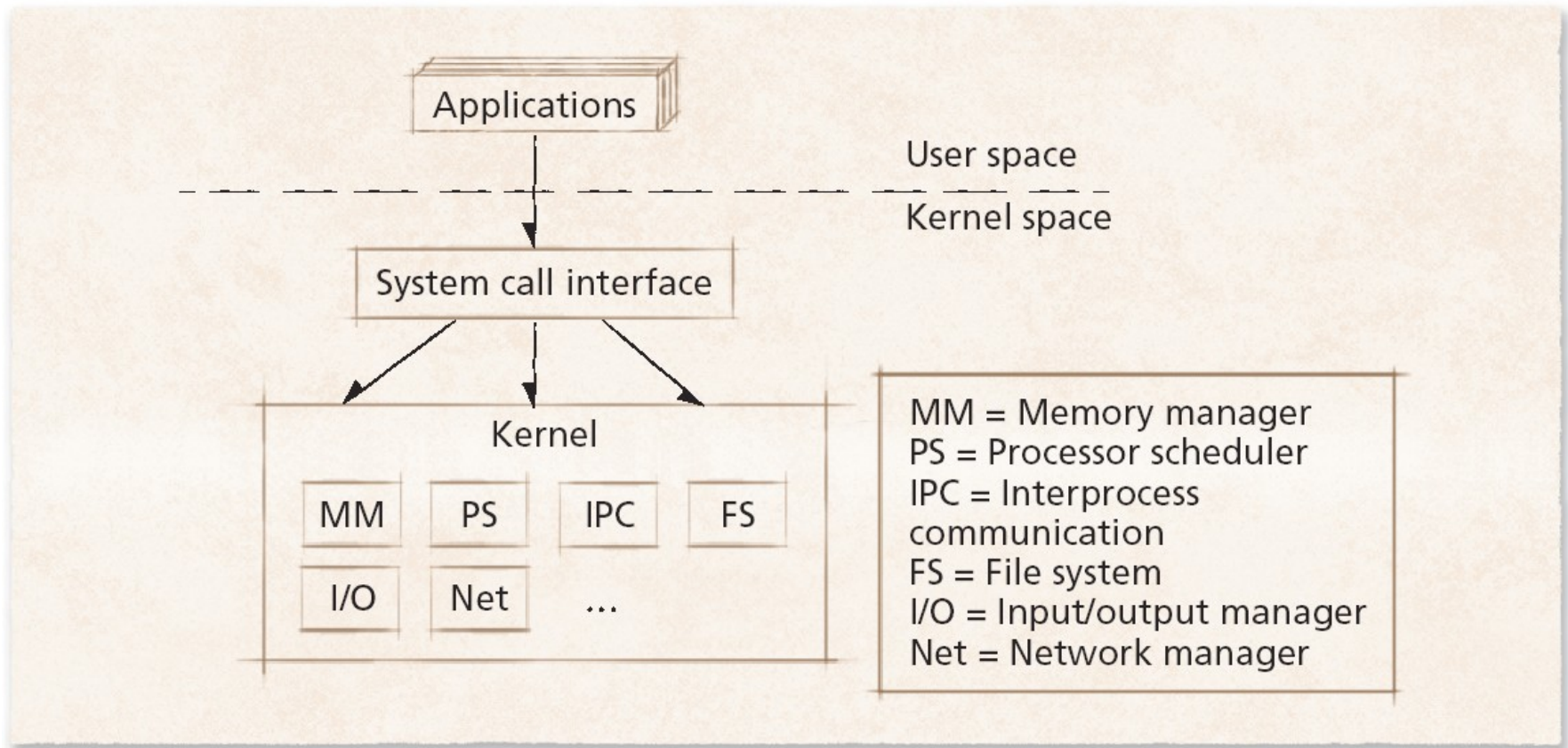
1.13.1 Monolithic Architecture

- Monolithic operating system
 - Every component contained in kernel
 - Any component can directly communicate with any other
 - Tend to be highly efficient
 - Disadvantage is difficulty determining source of subtle errors



1.13.1 Monolithic Architecture

Figure 1.3 Monolithic operating system kernel architecture.



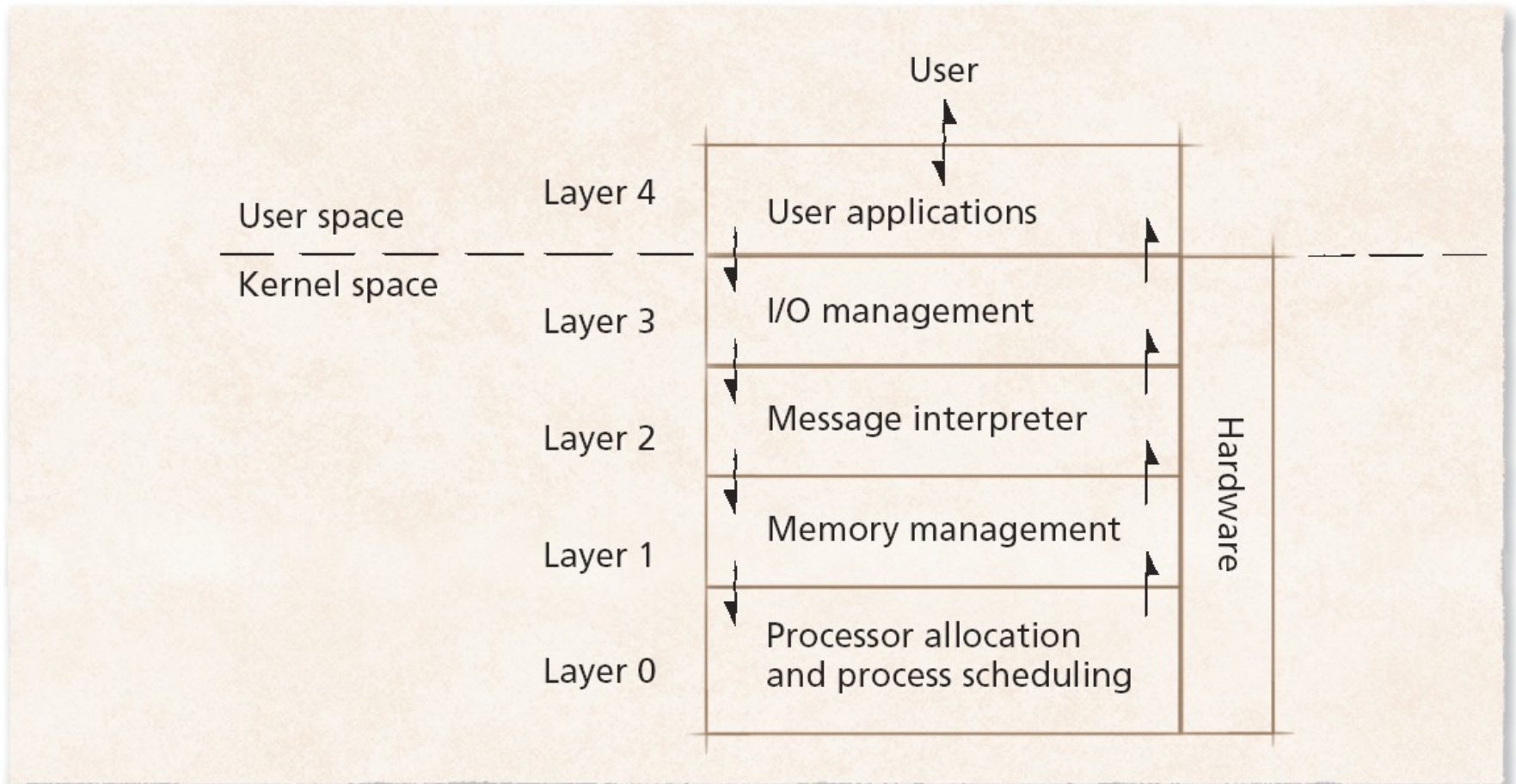
1.13.2 Layered Architecture

- Layered approach to operating systems
 - Tries to improve on monolithic kernel designs
 - Groups components that perform similar functions into layers
 - Each layer communicates only with layers immediately above and below it
 - Processes' requests might pass through many layers before completion
 - System throughput can be less than monolithic kernels
 - Additional methods must be invoked to pass data and control



1.13.2 Layered Architecture

Figure 1.4 Layers of the THE operating system.



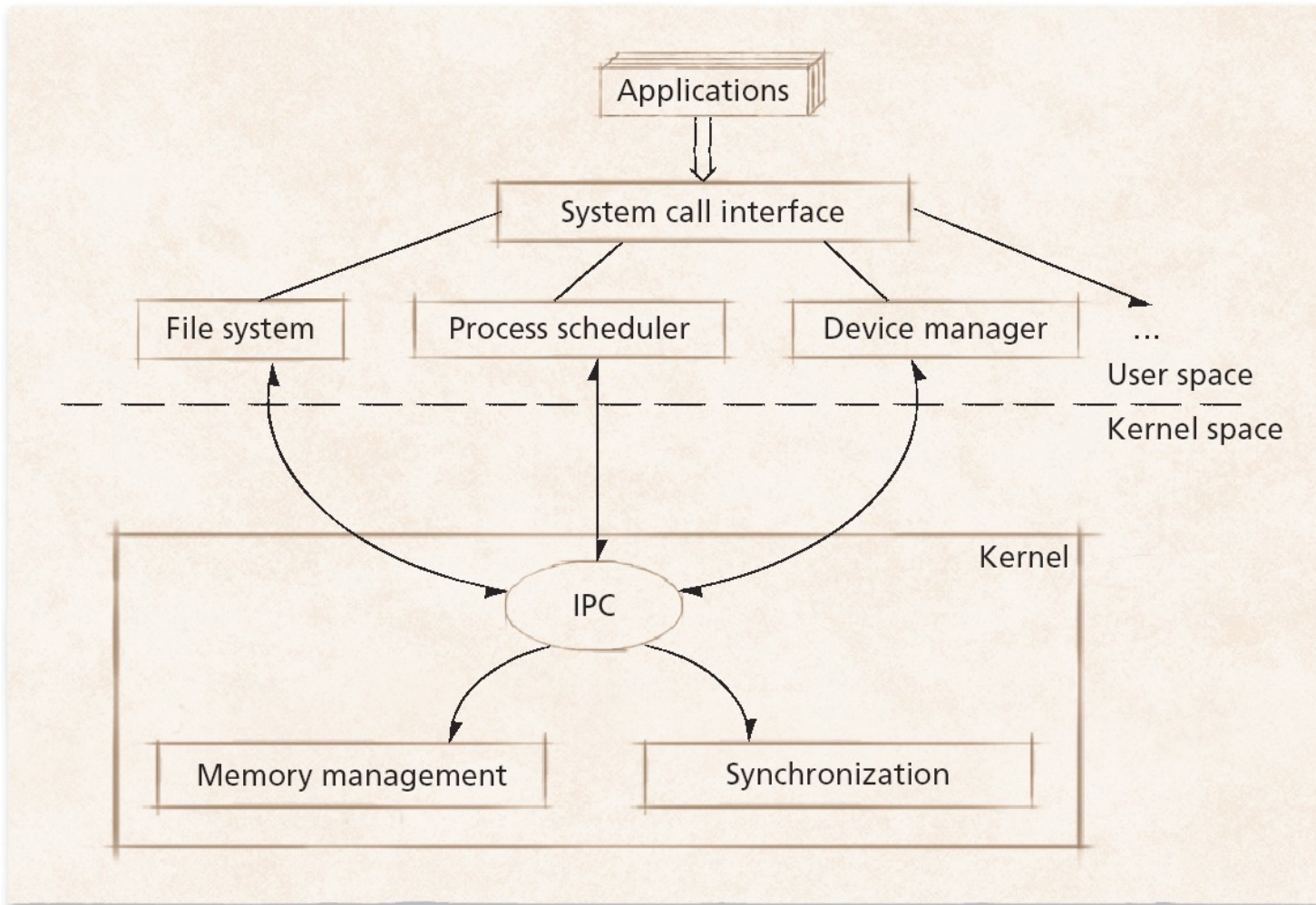
1.13.3 Microkernel Architecture

- Microkernel operating system architecture
 - Provides only small number of services
 - Attempt to keep kernel small and scalable
 - High degree of modularity
 - Extensible, portable and scalable
 - Increased level of intermodule communication
 - Can degrade system performance



1.13.3 Microkernel Architecture

Figure 1.5 Microkernel operating system architecture.



1.13.4 Networked and Distributed Operating Systems

- Network operating system
 - Runs on one computer
 - Allows its processes to access resources on remote computers
- Distributed operating system
 - Single operating system
 - Manages resources on more than one computer system
 - Goals include:
 - Transparent performance
 - Scalability
 - Fault tolerance
 - Consistency



1.13.4 Networked and Distributed Operating Systems

Figure 1.6 Client/server networked operating system model.

