8th
EDITION

understanding Operating Systems

Ann McIver McHoes
Ida M. Flynn

CENGAGE brain .com
Buy. Rent. Access.

Access student data files and other study tools on **cengagebrain.com**.

For detailed instructions visit
http://s-solutions.cengage.com/ctdownloads/

Store your Data Files on a USB drive for maximum efficiency in organizing and working with the files.

Macintosh users should use a program to expand WinZip or PKZip archives. Ask your instructor or lab coordinator for assistance.

Understanding Operating Systems

Eighth Edition

Ann McIver McHoes

Ida M. Flynn

CENGAGE Learning®

Australia • Canada • Mexico • Singapore • Spain • United Kingdom • United States

Understanding Operating Systems, Eighth Edition
Ann McIver McHoes & Ida M. Flynn

Senior Product Manager: Kathleen McMahon

Product Team Leader: Kristin McNary

Associate Product Manager: Kate Mason

Associate Content Development Manager: Alyssa Pratt

Production Director: Patty Stephan

Senior Content Project Manager: Jennifer Feltri-George

Manufacturing Planner: Julio Esperas

Art Director/Cover Design: Diana Graham

Production Service/Composition: SPi Global

Cover Photos: sumkin/Shutterstock.com

For product information and technology assistance, contact us at
Cengage Learning Customer & Sales Support, 1-800-354-9706

For permission to use material from this text or product,
submit all requests online at **www.cengage.com/permissions**
Further permissions questions can be emailed to
permissionrequest@cengage.com

Library of Congress Control Number: 2016962900

ISBN: 978-1-305-67425-7

Cengage Learning
20 Channel Center Street
Boston, MA 02210
USA

Cengage Learning is a leading provider of customized learning solutions with employees residing in nearly 40 different countries and sales in more than 125 countries around the world. Find your local representative at **www.cengage.com.**

Cengage Learning products are represented in Canada by Nelson Education, Ltd.

To learn more about Cengage Learning Solutions, visit **www.cengage.com**

Purchase any of our products at your local college store or at our preferred online store **www.cengagebrain.com**

Printed in the United States of America

Print Number: 01 Print Year: 2017

Dedicated to two inspiring colleagues:

Ida Moretti Flynn, award-winning teacher and a wonderful friend; her love for teaching lives on.

Bob Kleinmann, superb editor and soul mate – not in that order.

AMM

Contents

Appendix

Preface

Is this book for you? In these pages, we explain a very technical subject in a not-so-technical manner, putting the concepts of operating systems into words that many readers can quickly grasp.

For those who are new to the subject, this text demonstrates what operating systems are, what they do, how they do it, how their performance can be evaluated, and how they compare with each other. Throughout the textbook we describe the overall function of many unseen parts of the operating system and lead readers to additional resources where they can find more detailed information, if they so desire.

For readers with more technical backgrounds, this text introduces the subject concisely, describing the complexities of operating systems without going into intricate detail. One might say this book leaves off where other operating system textbooks begin.

To do so, we've made some assumptions about our audiences. First, we assume the readers have some familiarity with computing systems. Second, we assume they have a working knowledge of how to use an operating system and how it interacts with them. We recommend (although we don't require) that readers be familiar with at least one operating system. In the few places where, in previous editions, we used pseudocode to illustrate the inner workings of the operating systems, that code can be found in the Appendix. By moving these algorithms out of individual chapters, we have simplified our explanations of some complex events.

Although it is more difficult to understand how operating systems work than to memorize the details of a single operating system, gaining this understanding is a longer-lasting achievement, paying off in the long run because it allows one to adapt as technology changes—as, inevitably, it does.

Therefore, regardless of the level of expertise that the reader brings to the subject, the purpose of this book is to give computer users a solid background in the basics of operating systems, their functions and goals, and how they interact and interrelate.

Structure and Features

The organization of this book addresses a recurring problem with textbooks about technologies that continue to change—constant advances in evolving subject matter can make textbooks immediately outdated. To address this problem, our material is divided into two parts: first, the concepts, which do not change quickly, and second, the specifics of operating systems, which change dramatically over the course of years and even months. Our goal is to give readers the ability to apply their knowledge year after year, realizing that, although a command, or series of commands, used by one operating system may be different from another, the goals are the same and the functions of competing operating systems are also the same. It is for that reason, that we have structured this book in two parts.

Part One (the first 12 chapters) describes the concepts of operating systems by concentrating on several "managers" in turn, and then describing how these managers work together. In addition, Part One introduces network organization concepts, security, ethics, and system management.

Part Two examines actual operating systems: how they apply the theories presented in Part One and how they compare with each other.

Chapter 1 gives a brief introduction to the subject. The Memory Manager, described in Chapters 2 and 3, is the simplest component of the operating system to explain, and has been historically tied to the advances from one operating system to the next. We explain the role of the Processor (CPU) Manager in Chapters 4, 5, and 6, first discussing simple systems and then expanding the topic to include multiprocessing systems. By the time we reach the Device Manager in Chapter 7 and the File Manager in Chapter 8, readers will have been introduced to many key concepts found in every operating system. Chapters 9 and 10 introduce basic concepts related to networking. Chapters 11 and 12 discuss security, ethics, and system management, including some of the tradeoffs that operating systems designers consider when attempting to satisfy the needs of their user population.

In Part Two we explore four operating systems in the order of their first release: UNIX, Windows, Linux, and Android. Here, each chapter includes a discussion describing how that operating system applies the concepts discussed in Part One. Again, we must stress that this is a general discussion—an in-depth examination of an operating system would require details based on its current standard version, which can't be done in a textbook. We strongly suggest that readers use our discussion as a guide—a base to work from—when comparing the advantages and disadvantages of a specific operating system, and supplement our work with current academic research, which is readily available online.

Each chapter includes learning objectives, key terms, research topics, exercises, and a spotlight on industry experts who have left their mark in computer science. For technically oriented readers, the exercises at the end of each chapter include some problems for advanced students. Please note that these advanced exercises assume knowledge of matters not presented in the book, but they're good for anyone who enjoys a challenge. We expect some readers who are new to the subject will cheerfully pass them by.

The text concludes with several reference aids. Within each chapter, important terms are listed at its conclusion as key terms. The Windows chapter also includes a table that briefly lists all acronyms or abbreviations used in that chapter. An extensive end-of-book Glossary includes brief reader-friendly definitions for hundreds of terms used in these pages; note that this glossary is specific to the way these terms are used in this textbook. The Bibliography can guide the reader to basic research on the subject. Finally, the Appendix features pseudocode algorithms referenced in several chapters, and a section of the ACM Code of Ethics.

In an attempt to bring the concepts closer to home, throughout the book we've added real-life examples to illustrate abstract concepts. However, let no one confuse our conversational style with our considerable respect for the subject matter. The subject of operating systems is a complex one and it cannot be covered completely in these few pages. Therefore, in this textbook we do not attempt to give an in-depth treatise of operating systems theory and applications. This is an overall view.

Not included in this text is a detailed discussion of databases and data structures, except as they are used to resolve process synchronization problems, or the work of specific operating systems. This is because these structures only tangentially relate to operating systems and are frequently the subject of other courses. We suggest that readers begin by learning the basics as presented in the following pages and pursue these complex subjects in their future studies.

Changes to this Edition

This edition has been thoroughly updated and features many improvements over previous editions:

- Renewed emphasis on the role of the talented people who designed and wrote operating systems, as well as their design decisions, which can affect how the resulting system works.
- Added more screenshots from a variety of operating systems, including Macintosh OS (which runs UNIX), Windows, Android phone and tablet, and Linux.
- Expanded our discussions of cloud computing and cloud storage.
- Revised networking discussions to reflect emerging designs and technology.
- Retained an emphasis on student understanding and original thinking in the exercises, rather than on memorization or cut-and-paste facts. This is because our book's answer key is often available online shortly after publication, so in these pages we have routinely asked students to use their own words to explain concepts.

- Expanded cross-references from Part Two to the concepts taught in Part One to help students link specific system features with the concepts discussed in the beginning chapters.
- Added emphasis on available command-mode options in each operating system for readers who want to explore their system more directly, without using the menus.
- Included online resources for more information about many of the highly technical subjects introduced in this text. Please remember that in the field of computer science, online links go bad frequently, but by providing these links to our readers, they will have a good starting place from which they can search for more current info.
- Updated artwork and references to the expanding influence of wireless technology.
- Removed examples in assembly language, which is not widely studied in introductory classes, and replaced them with pseudocode and prose descriptions.

Numerous other changes throughout the text include editorial clarifications, expanded captions, and improved illustrations.

A Note for Instructors

The following supplements are available when this text is used in a classroom setting. All supplements can be downloaded from the Instructor Companion Site. Simply search for this text at sso.cengage.com. An instructor login is required.

Instructor's Manual. The Instructor's Manual that accompanies this textbook includes additional instructional material to assist in class preparation, including Sample Syllabi, Chapter Outlines, Technical Notes, Lecture Notes, Quick Quizzes, Teaching Tips, and Discussion Topics.

Test Bank. Cengage Testing Powered by Cognero is a flexible, online system that allows you to:

- author, edit, and manage test bank content from multiple Cengage solutions;
- create multiple test versions in an instant;
- deliver tests from your LMS, your classroom, or wherever you want.

PowerPoint Presentations. This book comes with Microsoft PowerPoint slides for each chapter. These are included as a teaching aid for classroom presentations, either to make available to students on the network for chapter review, or to be printed for classroom distribution. Instructors can add their own slides for additional topics that they wish to introduce to the class.

Solutions. Selected solutions to Exercises are provided.

Order of Presentation

We have built this text with a modular construction to accommodate several alternative sequences, depending on the instructor's preference.

- For example, the syllabus can follow the chapters as listed from Chapter 1 through Chapter 12 to present the core concepts that all operating systems have in common. Using this path, students will learn about the management of memory, processors, devices, files, and networks, in that order.
- An alternative path might begin with Chapter 1, move next to processor management in Chapters 4 through 6, then to memory management in Chapters 2 and 3, touch on systems security and management in Chapters 11 and 12, and finally move to device and file management in Chapters 7 and 8. Because networking is often the subject of another course, instructors may choose to bypass Chapters 9 and 10, or include them for a more thorough treatment of operating systems.

We hope you find our discussion of ethics helpful in Chapter 11, which is here in response to requests by university adopters of the text who asked us to include this subject, even though it is sometimes the subject of a separate course.

When teaching one or more operating systems from Part Two, keep in mind that we structured each of these four chapters the same way we presented concepts in the first 12 chapters. That is, they discuss the management of memory, processors, files, devices, networks, and systems, in that order, with a special section demonstrating the user interfaces for each operating system. To illustrate the use of graphical user interfaces in UNIX systems, we include screenshots from the Macintosh OS X operating system.

By including the Android operating system, which is specifically designed for use in a mobile environment using phones and tablets, we are able to explore the challenges unique to these computing situations.

Acknowledgments

Our gratitude goes to all of our friends and colleagues who were so generous with their encouragement, advice, and support over the two decades of this publication. Special thanks go to Bob Kleinmann, Eleanor Irwin, and Roger Flynn for their assistance.

As always, thanks to those at Cengage, Brooks/Cole, and PWS Publishing who have made significant contributions to all eight editions of this text, especially Alyssa Pratt, Kallie Swanson, Mike Sugarman, and Mary Thomas Stone.

And to the many students and instructors who have sent helpful comments and suggestions since publication of our first edition in 1991, we thank you. Please keep them coming.

Ann McIver McHoes, mchoesa@duq.edu

Ida Moretti Flynn (1945–2004)

Part One

Operating Systems Concepts

This text explores the core mechanisms of operating systems, which manage a computing system's hardware and software. That includes its memory, processing capability, devices, files, and networks—and how to do all of this in an appropriate and secure fashion. Here, in Part One, we present an overview of an operating system's essentials.

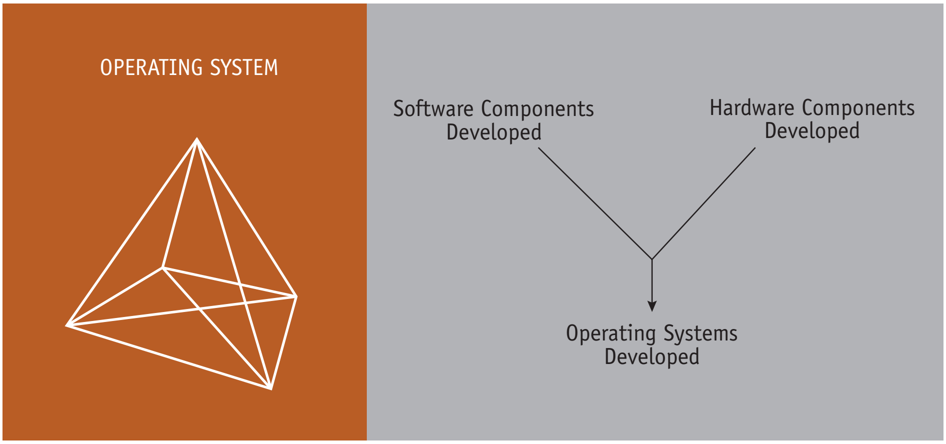
- Chapter 1 introduces the subject of operating systems.
- Chapters 2 and 3 discuss the management of main memory resources.
- Chapters 4 through 6 cover single processor and multiprocessor management.
- Chapter 7 concentrates on managing available devices without conflicts.
- Chapter 8 is devoted to the management of files, including those that hold system instructions as well as your data.
- Chapters 9 and 10 briefly review operating systems for networks.
- Chapter 11 discusses system security.
- Chapter 12 explores system management.

In Part Two (Chapters 13 through 16), we look at four specific operating systems and how they apply the overall concepts presented in the first 12 chapters.

Throughout our discussion of this very technical subject, we try to include definitions of terms that might be unfamiliar, but it isn't always possible to describe a function and define the technical terms while keeping the explanation clear. Therefore, we've put the key terms with definitions at the end of each chapter, as well as in the glossary at the end of the text. Items listed in the Key Terms are shown in **boldface** the first time they are mentioned significantly.

Throughout this book we keep our descriptions and examples as simple as possible to introduce the system's complexities without getting bogged down in technical detail. Therefore, remember that for almost every topic explained in the following pages, there's much more information that's readily available for study. Our goal is to introduce the subject and to encourage our readers to independently pursue topics of special interest. Enjoy.

Chapter 1 | Introducing Operating Systems



"I think there is a world market for maybe five computers."

—Attributed to Thomas J. Watson (1874–1956; chairman of IBM 1949–1956)

Learning Objectives

After completing this chapter, you should be able to describe:

- How operating systems have evolved through the decades
- The basic role of an operating system
- How operating system software manages it subsystems
- The role of computer system hardware on the development of its operating system
- How operations systems are adapted to serve batch, interactive, real-time, hybrid, and embedded systems
- How operating systems designers envision their role and plan their work

To understand an **operating system** is to begin to understand the workings of an entire computer system, because the operating system software manages each and every piece of hardware and software. In the pages that follow, we explore what operating systems are, how they work, what they do, and why.

This chapter briefly describes the workings of operating systems on the simplest scale. The following chapters explore each component in more depth, and show how its function relates to the other parts of the operating system. In other words, we see how the pieces work together harmoniously to keep the computer system working smoothly.

What Is an Operating System?

A computer system typically consists of software (programs) and hardware (the tangible machine and its electronic components). The operating system is the most important software— it's the portion of the computing system that manages all of the hardware and all of the other software. To be specific, the operating system software controls every file, every device, every section of main memory, and every moment of processing time. It controls who can use the system and how. In short, the operating system is the boss.

Therefore, each time the user sends a command, the operating system must make sure that the command is executed; or, if it's not executed, it must arrange for the user to get a message explaining the error. This doesn't necessarily mean that the operating system executes the command or sends the error message, but it does control the parts of the system that do.

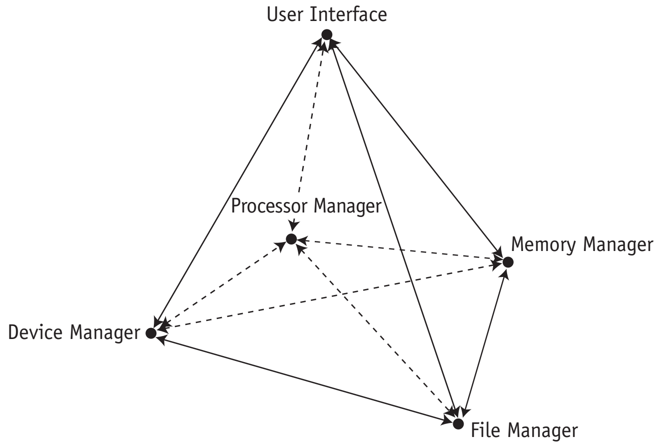
Operating System Software

The pyramid shown in Figure 1.1 is an abstract representation of the operating system in its simplest form, and demonstrates how its major components typically work together.

At the base of the pyramid are the four essential managers of every major operating system: **Memory Manager**, **Processor Manager**, **Device Manager**, and **File Manager**. These managers, and their interactions, are discussed in detail in Chapters 1 through 8 of this book. Each manager works closely with the other managers as each one performs its unique role. At the top of the pyramid is the User Interface, which allows the user to issue commands to the operating system. Because this component has specific elements, in both form and function, it is often very different from one operating system to the next—sometimes even between different versions of the same operating system.

(figure 1.1)

This pyramid represents an operating system on a stand-alone computer unconnected to a network. It shows the four subsystem managers and the User Interface.

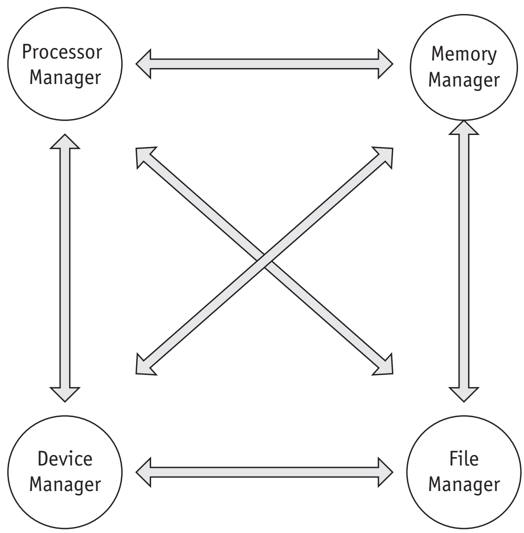


Regardless of the size or configuration of the system, the four managers, illustrated in Figure 1.2, must, at a minimum, perform the following tasks while collectively keeping the system working smoothly:

- Monitor the system's resources
- Enforce the policies that determine what component gets what resources, when, and how much
- Allocate the resources when appropriate
- Deallocate the resources when appropriate

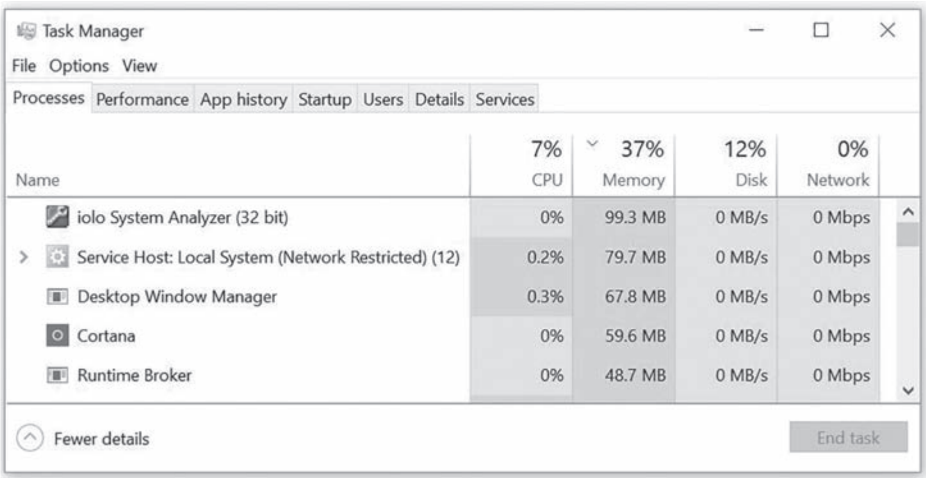
(figure 1.2)

Each manager at the base of the pyramid takes responsibility for its own tasks while working harmoniously with every other manager.



For example, the Memory Manager must keep track of the status of the computer system's main memory space, allocate the correct amount of it to incoming processes, and deallocate that space when appropriate—all while enforcing the policies that were established by the designers of the operating system.

An additional management task, networking, has not always been an integral part of operating systems. Today the vast majority of major operating systems incorporate a **Network Manager**, see Figure 1.3, to coordinate the services required for multiple systems to work cohesively together. For example, the Network Manager must coordinate the workings of the networked resources, which might include shared access to memory space, processors, printers, databases, monitors, applications, and more. This can be a complex balancing act as the number of resources increases, as it often does.



(figure 1.3)

The Windows 10 Task Manager displays a snapshot of the system's CPU, main memory, disk, and network activity.

Main Memory Management

The Memory Manager (the subject of Chapters 2 and 3) is in charge of main memory, widely known as **RAM** (short for random access memory). The Memory Manager checks the validity of each request for memory space, and if it is a legal request, allocates a portion of memory that isn't already in use. If the memory space becomes fragmented, this manager might use policies established by the operating system's designers to reallocate memory to make more useable space available for other jobs that are waiting. Finally, when the job or process is finished, the Memory Manager deallocates its allotted memory space.

A key feature of RAM chips—the hardware that comprises computer memory—is that they depend on the constant flow of electricity to hold data. If the power fails or is turned off, the contents of RAM is wiped clean. This is one reason why computer system

RAM stands for random access memory and is the computer's main memory. It's sometimes called "primary storage" to distinguish it from "secondary storage," where data is stored on hard drives or other devices.

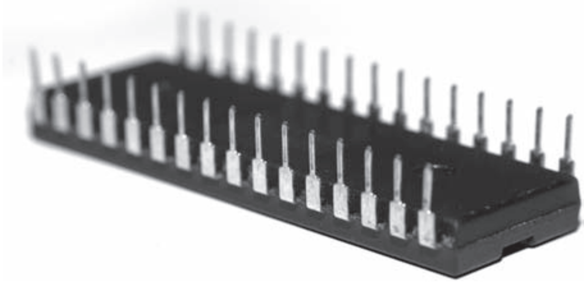
designers attempt to build elegant shutdown procedures, so that the contents of RAM can be stored on a nonvolatile device, such as a hard drive, before the main memory chips lose power during computer shutdown.

A critical responsibility of the Memory Manager is to protect all of the space in main memory, particularly the space occupied by the operating system itself—it can't allow any part of the operating system to be accidentally or intentionally altered because that would lead to instability or a system crash.

Another kind of memory that's critical when the computer is powered on is read-only memory (often shortened to ROM), shown in Figure 1.4. This ROM chip holds software called **firmware**: the programming code that is used to start the computer and perform other necessary tasks. To put it in simplest form, it describes, in prescribed steps, when and how to load each piece of the operating system after the power is turned on, up to the point that the computer is ready for use. The contents of the ROM chip are nonvolatile, meaning that they are not erased when the power is turned off, unlike the contents of RAM.

(figure 1.4)

A computer's relatively small ROM chip contains the firmware (unchanging software) that prescribes the system's initialization every time the system's power is turned on.



Processor Management

The Processor Manager (discussed in Chapters 4 through 6) decides how to allocate the central processing unit (CPU); an important function of the Processor Manager is to keep track of the status of each job, process, thread, and so on. We will discuss all of these in the chapters that follow, but for this overview, let's limit our discussion to a **process** and define it as a program's "instance of execution." A simple example could be a request to solve a mathematical equation: This would be a single job consisting of several processes, with each process performing a part of the overall equation.

The Processor Manager is required to monitor the computer's CPU to see if it's busy executing a process or sitting idle as it waits for some other command to finish execution. Generally, systems are more efficient when their CPUs are kept busy. The Processor

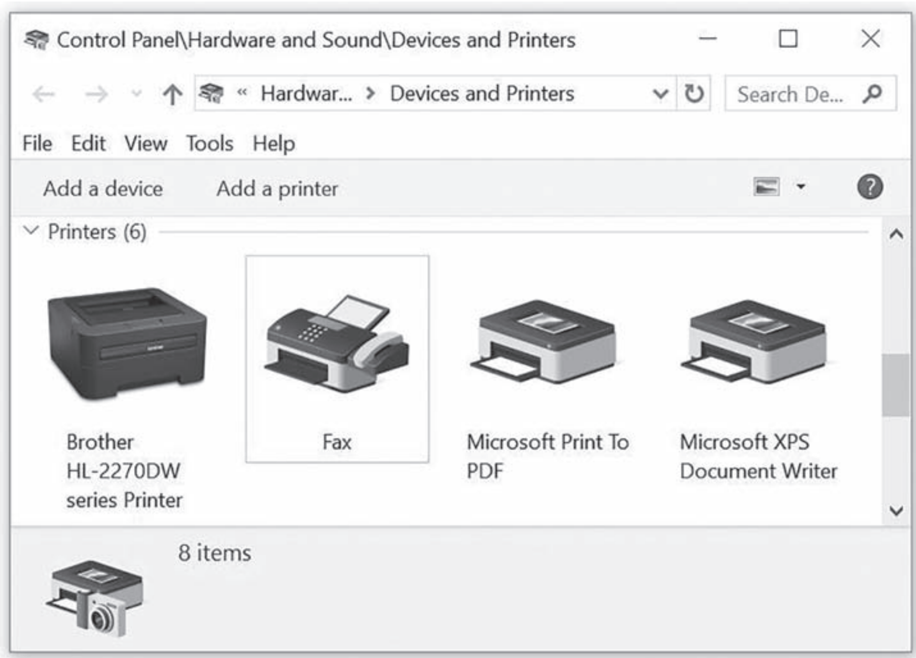
Manager handles each process's transition, from one state of execution to another, as it moves from the starting queue, through the running state, and, finally, to the finish line (where it then tends to the next process). Therefore, this manager can be compared to a traffic controller. When the process is finished, or when the maximum amount of computation time has expired, the Processor Manager reclaims the CPU so it can allocate it to the next waiting process. If the computer has multiple CPUs, as with a multicore system, the Process Manager's responsibilities are greatly complicated.

Device Management

The Device Manager (the subject of Chapter 7) is responsible for connecting with every device that's available on the system, and for choosing the most efficient way to allocate each of these printers, ports, disk drives, and more, based on the device scheduling policies selected by the designers of the operating system.

A flash memory device is an example of secondary storage because it doesn't lose data when its power is turned off. Still, some operating systems allow users to plug in such a device to improve the performance of main memory.

Good device management requires that this part of the operating system uniquely identify each device, start its operation when appropriate, monitor its progress, and, finally, deallocate the device to make the operating system available to the next waiting process. This isn't as easy as it sounds because of the exceptionally wide range of devices that can be attached to any system, such as the system shown in Figure 1.5.



(figure 1.5)

This computer, running the Windows 10 operating system, has device drivers loaded for all of the printing devices shown here.

For example, let's say you're adding a printer to your system. There are several kinds of printers commonly available (laser, inkjet, inkless thermal, etc.) and they're made by manufacturers that number in the hundreds or thousands. To complicate things, some devices can be shared, while some can be used by only one user or one job at a time. Designing an operating system to manage such a wide range of printers (as well as monitors, keyboards, pointing devices, disk drives, cameras, scanners, and so on) is a daunting task. To do so, each device has its own software, called a device driver, which contains the detailed instructions required by the operating system to start that device, allocate it to a job, use the device correctly, and deallocate it when it's appropriate.

File Management

The File Manager (described in Chapter 8), keeps track of every file in the system, including data files, program files, utilities, compilers, applications, and so on. By following the access policies determined by the system designers, the File Manager enforces restrictions on who has access to which files. Many operating systems allow authorized individuals to change these permissions and restrictions. The File Manager also controls the range of actions that each user is allowed to perform on the files after they access them. For example, one user might have read-only access to a critical database, while the systems administrator might hold read-and-write access with the authority to create and delete files in the same database. Access control is a key part of good file management and is tightly coupled with system security software.

When the File Manager allocates space on a secondary **storage** device, such as a hard drive, flash drive, archival device, and so on, it must do so knowing the technical requirements of that device. For example, if it needs to store an archival copy of a large file, it needs to know if the device stores it more efficiently as one large block or in several smaller pieces that are linked through an index. This information is also necessary for the file to be correctly retrieved later. Later, if this large file must be modified after it has been stored, the File Manager must be capable of making those modifications as accurately and efficiently as possible.

Network Management

Operating systems with networking capability have a fifth essential manager called the Network Manager (the subject of Chapters 9 and 10) that provides a convenient way for authorized users to share resources. To do so, this manager must take overall responsibility for every aspect of network connectivity, including the requirements of the available devices as well as files, memory space, CPU capacity, transmission connections, and types of encryption (if necessary). Networks with many available resources require management of a vast range of alternative elements, which enormously complicates the tasks required to add network management capabilities.