An Introduction to Windows Operating System

Einar Krogh

bookboon
The eBook company

EINAR KROGH

AN INTRODUCTION TO WINDOWS OPERATING SYSTEM

An Introduction to Windows Operating System
2nd edition

ISBN 978-87-403-1935-4
Peer review by Høgskolelektor Lars Vidar Magnusson, Høgskolen i Østfold

CONTENTS

INTRODUCTION

This is a book about the Windows operating system written for courses on operating systems.

Students of operating systems need some knowledge of the Windows operating system. This book presents the key components of the Windows operating system in a short and straightforward way and introduces how a computer system using the Windows operating system works.

The book also discusses general theory of operating systems. Central subjects in the book are how applications are running on a computer and how applications communicate with hardware.

The book is both suitable for introductory courses on the architecture and internals of the Windows operating system and for beginning courses on operating systems in general.

1 ABOUT WINDOWS HISTORY

A Personal Computer (PC) is a computer designed for use by private individuals. The typical use of a PC has been for word processing, spreadsheets, databases, Web browsing, e-mail and computer games.

The history of personal computers began in earnest in 1977, with the introduction of the microcomputer. When the microprocessor was developed, it was possible to create computers so affordable to purchase that private individuals could buy them.

When IBM (International Business Machines) began producing a Personal Computer in 1980 it created a need for an operating system for this new computer. IBM approached Microsoft with a request. Microsoft was not developing operating systems at this time, but since the mid-1970s had been a leader in the development of tools for the programming language, BASIC. Microsoft acquired an operating system by purchasing it from Seattle Computer Products, and then made changes to this system so that it was suited for PCs. Microsoft called the operating system MS-DOS.

1.1 MS-DOS

The new operating system got the name MS-DOS 1.0

DOS = Disk Operating System

MS-DOS was an operating system based on a simple communication between user and computer. The user entered commands on the screen using the keyboard, as a mouse was used to a limited extent at this point.

MS-DOS was the most common operating system for PCs before Windows, and was the base operating system in the first versions of Windows. MS-DOS therefore continued to live long after the introduction of Windows.

Microsoft has made many versions of MS-DOS, with MS-DOS 8.0 being the latest version, which was launched in the year 2000.

1.2 THE FIRST VERSIONS OF WINDOWS

The development of operating systems with a graphical user interface started in the 1980s, which was when Microsoft decided to give MS-DOS a graphical user interface. Microsoft created a graphical program for this purpose called Windows. At first, Windows was not a separate operating system, but instead a graphical application that used MS-DOS as operating system.

Windows 1.0 was the first version and came in 1983, though the first versions of Windows (1.0 and 2.0) were not a success. The reason for this was that there was little software for Windows, while the existing software was unstable and simple. It was not until Windows 3.0 and then Windows 3.1 that Windows became widely taken into use.

Version	*Release*	*About the version*
Windows 3.0	1990	Ten million copies sold.
Windows 3.1	1991	Windows becomes widespread.
Windows 95	1995	New and improved version.
Windows 98	1998	Integrating the Internet.
Windows Millennium	2000	Focus on multimedia.

Figure 1.1: The figure shows the first versions of Windows

The operating systems Windows 95, Windows 98 and Windows Millennium were quite similar, so they were therefore called Windows 9x.

Windows 95

Windows 95 was a major improvement from earlier versions of Windows, central to this being an enhanced usability and better networking capabilities. The new features in Windows 95 were multitasking and the automatic detection and configuration of equipment (Plug and Play).

Windows 98

Windows 98 was an upgrade and improvement of Windows 95, as Microsoft wanted to implement the Internet in Windows at this time. As a result, Windows 98 contained Internet Explorer and other programs for the Internet.

Windows Millennium

Windows Millennium was the last release in the series based on the Windows 9x platform, and was aimed at the domestic market with a focus on multimedia.

1.3 WINDOWS NT

Computers acquired more and more memory, a higher processor speed and more disk space. MS-DOS could not handle a lot of memory, and could not run multiple applications simultaneously, so there was a need for a new operating system. This led to the developing of Windows NT, in which NT stands for New Technology.

Microsoft released the first version of Windows NT in 1993 and called it Windows NT 3.1, because it came at the time when Windows 3.1 was in use. Windows NT 3.1 had the same appearance and user interface as Windows 3.1, but they were completely different operating systems.

Windows NT 4.0 came in 1996, with the two basic versions of Windows NT 4.0 being:

- Windows NT Server
- Windows NT Workstation

Windows NT 4.0 Server was a network operating system designed to be used on servers in local networks. Windows NT 4.0 Workstation was designed for use on both home computers and workstations, but was primarily used by companies as workstations in local networks.

Windows NT 4.0 was an advanced 32-bit operating system designed to be secure, stable and flexible, thus making it possible to use multiple processors on the same computer. Windows NT 4.0 could utilize a large memory and large hard drives, which meant that the operating system was well scalable.

Windows NT introduced users as a part of the system's security model, so in order to use a workstation in a modern Windows system you must be a registered user. One can define multiple users on a workstation, and each user has certain rights to files and to access system resources in general.

1.4 WINDOWS VERSIONS BASED ON WINDOWS NT

Windows NT operating system is the basis for the current versions of Windows.

Version	*Release*	*About the version*
Windows 2000	2000	Was not intended for home computers.
Windows XP	2001	Both for home computers and workstations.
Windows Vista	2006	New user interface with Windows Aero.
Windows 7	2009	Emphasis on functionality and performance.
Windows 8	2012	New graphical interface Microsoft Metro.
Windows 10	2015	Better functionality between different classes of device.

Figure 1.2: The figure shows the versions of Windows based on Windows NT

Windows NT has become a widespread operating system in the computer world, as today you can find variations of the original Windows NT on laptops, desktops, servers and Xbox consoles worldwide.

Windows 2000

Windows 2000 (Windows NT 5.0) was built on Windows NT 4.0. In addition, Windows 2000 had most of the useful qualities of Windows 98, such as support for Plug and Play. Windows 2000's operating system was available in several versions. One version was for workstations, and there were several versions for servers.

Windows XP

Launched in August 2001, Windows XP (Windows NT 5.1) has been the most popular version of Windows, based on the number of copies sold.

Windows Vista

Launched in November 2006, Windows Vista (Windows NT 6.0) contained hundreds of new and revised features.

Windows 7

Windows 7 (Windows NT 6.1) came in October 2009. Unlike previous versions of Windows, Windows 7 did not contain many new features, basically being more of an upgrade of Windows Vista. The objective of Windows 7 was an operating system that had increased functionality and performance over previous versions.

Windows 8

Windows 8 (Windows NT 6.2) came in October 2012, and contained a new graphical interface known as Metro, which is used for tablets, laptops, desktops and Windows Phone. Optimized for touch screens, Metro can also be controlled with a mouse and keyboard.

Windows 10

Windows 10 (Windows NT 10.0) came in July 2015. Windows 10 is intended as the last version as there will be no Windows 11. Instead Windows Update will update the existing Windows 10 to new versions. At this point there have been three such updates, one in November 2015 (version 1511), one in July 2016 (version 1607) and one in Mars 2017 (version 1703).

Microsoft described Windows 10 as an *operating system as a service* that would receive ongoing updates to its features and functionality. Windows 10 harmonizes the user experience and functionality between different classes of device, and addresses shortcomings in the user interface that were introduced in Windows 8.

Windows 10 has become a shared platform known as OneCore, and it runs on PCs, phones, the Xbox One game console, the HoloLens and Internet of Things (IoT) devices such as Raspberry Pi 2.

1.5 WINDOWS SERVER

Microsoft has made several versions of the Windows operating system adapted to act as a server in local networks.

Microsoft's first attempt at creating an operating system with network features was Windows 3.11 (1992). Another name for Windows 3.11 is Windows for Workgroups, and this version had some additional features that gave network support:

- Network cards and cables
- Sharing directories, disks and printers
- E-mail and instant messaging functionality

However, it was with Windows NT that Microsoft began developing a network operating system intended to act as a server.

Version	Release
Windows NT 4.0 Server	1996
Windows Server 2000	2000
Windows Server 2003	2003
Windows Server 2008	2008
Windows Server 2012	2012
Windows Server 2016	2016

Figure 1.3: The figure shows the server versions of Windows

Windows NT 4.0 Server

It was first with Windows NT 4.0 that Microsoft got a proper network operating system adapted to the task as a server in a local area network, as the Windows NT 4.0 Server could serve many users in a network.

Windows Server 2000

The sequel to Windows NT 4.0 Server was Windows Server 2000. The main novelty in Windows Server 2000 was Active Directory, which is Microsoft's directory service for managing users and resources in a local area network.

Windows Server 2003

Built on Windows XP, Windows Server 2003 had several improvements over Windows Server 2000. New in Windows Server 2003 was the Internet Information Services (IIS) used to create web pages for servers.

Windows Server 2008

Launched in February 2008, Windows Server 2008 was an upgrade of Windows Server 2003. Windows Server 2008 had enhancements to Active Directory, Group Policy, disk management and security.

Windows Server 2012

Windows Server 2012, also called Windows Server 8, debuted in September 2012. New with Windows Server 2012 was better support for server virtualization and the use of the cloud (Cloud Computing). Windows Server 2012 includes a new file system (ReFS), but this file system is only used by file servers.

Windows Server 2016

With Windows Server 2016, Microsoft aims to assist customers in modernizing on premise data centers, making it easier to move workloads out to its Azure public cloud. New in Windows Server 2016 is Nano Server. Nano Server is a scaled down, purpose-built operating system designed to run modern cloud applications and act as a platform for containers. Support for containers is another of the standout features for Windows Server 2016.

1.6 CONTROL QUESTIONS

1. At what point in time was Windows taken into use as an operating system?
2. What was the task of MS-DOS in the first versions of Windows?
3. Mention some reasons that Microsoft developed Windows NT.
4. Mention some differences between Windows NT and Windows 9x.
5. Mention some differences between Windows Server and Windows for PCs.
6. How is Windows 10 different from earlier versions of Windows?

2 THE TASKS OF AN OPERATING SYSTEM

There are two types of software running on a computer: software applications and system software. A software application is a program designed for end users, while system software consists of low-level programs belonging to the operating system, compilers and utilities for managing resources.

An operating system is software that enables services for software applications to run on a computer. An important task of an operating system is taking care of the communication between the software applications and hardware devices attached to your computer. For example, a word processor communicates with devices such as a keyboard and mouse.

Operating systems are large programs consisting of thousands of functions, which provide services of various kinds. Often called by events in the system, the functions perform a service when needed. To make operating systems fast and most effective, the functions are often written in C or C++, but also low-level programming languages such as Assembly has occasionally been used.

2.1 ABOUT THE CONSTRUCTION OF COMPUTERS

To understand the tasks of an operating system, it is necessary to have some knowledge of computer systems. In this text, there will be no detailed description of the constructions of computers, but we will look at some key components of the computer system and show why these need to be administered by an operating system.

Item	*Description*
Processor	A processor is the part of the computer system that is able to execute program code.
Memory	Computer memory is a temporary storage location for data and the program code. Data is lost in the memory when the power of the computer is turned off.
Hard drive	The task of hard drives is to permanently store data on a computer. The hard drives store program files, documents and data files.
Input/output devices	Equipment used for communication between the computer and a user. The most common input/output devices are the keyboard, mouse and screen.

Figure 2.1: The figure shows some key devices in a computer system

A processor executes the instructions that a computer program is comprised of. The processor is therefore the central unit of program execution.

Several applications are often simultaneously executed on a computer. It is therefore a task for the operating system to launch programs and allocate time for programs in one or more processors.

The applications running are loaded into a computer's memory for storage. Consequently, it is therefore necessary to keep track of where in the memory different data resides. In addition, there is a need to keep track of which parts of the memory are available and which parts are in use. Managing computer memory is a task for the operating system.

Files are stored on a hard disk. The operating system keeps track of where files on the hard disk are situated, as well as what parts of the hard disk are in use and what parts are not in use.

Input/output means communication between software applications and screen, keyboard, mouse and similar devices. The operating system takes care of the input/output to make it easier for applications to communicate with different drivers for input/output devices.

In addition to creating services for the computer system, it is a task for an operating system to provide a user interface for the computer. Windows has a graphical user interface in which the user uses the keyboard, mouse and touchscreen to interact with applications.

2.2 CENTRAL TASKS FOR AN OPERATING SYSTEM

We want fast and reliable computers; hence, an important task for an operating system is to utilize computer resources in the best possible way.

There are two main goals of an operating system:

- Managing applications and giving applications access to hardware.
- Manage data system resources optimally.

Important tasks for an operating system are to:

1. Manage applications running on the computer.
2. Take care of input and output.
3. Manage communication between software applications and hardware.
4. Manage computer memory.
5. Manage the file system.
6. Manage networking.
7. Take care of the security of the computer system.
8. Provide a user interface for the computer.

The operating system manages applications running on the computer. The operating system starts and stops applications, and provides time for applications in the processor. An important task for an operating system is to help facilitate communication between applications and hardware, while user applications access the hardware through the operating system.

An operating system is a layer between applications and hardware. A user interacts with an application, the application interacts with the operating system and the operating system communicates with hardware.

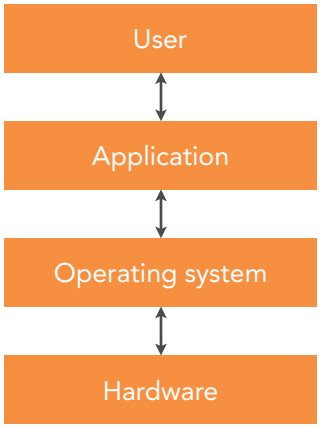

Figure 2.2: The interaction in a computer system

Hardware equipment consists of various types and uses different drivers, whereas the operating system takes care of the communication with the hardware. This makes it much easier to create computer programs since programmers do not need to consider communication with different drivers.

Modern operating systems use multitasking, which allows multiple applications to run simultaneously, thereby making it possible to perform more than one action on the computer at approximately the same time. To help achieve this, programs take turns running in the processor. An application executes in a processor for a short time before it must be out of the processor, and another application is loaded into the processor for execution.

Data security is important when using computer technology, particularly for computers attached to a network. The objective of computer security is therefore to provide information and data against unauthorized access, theft, destruction and natural disasters.

2.3 CONTROL QUESTIONS

1. Describe the tasks of the processor, memory, hard drive and input/output devices.
2. Explain why it is beneficial that an operating system manages the processor, memory, hard drive and input/output devices.
3. What makes it difficult for applications to communicate directly with hardware?
4. What are the two main goals for an operating system?
5. Mention the most important tasks for an operating system.

3 SOME CONCEPTS AND TERMS OF THE WINDOWS OPERATING SYSTEM

In this chapter, we get to know some basic concepts and terms of the Windows operating system.

3.1 WINDOWS API

Windows API is a library of functions that form an interface between the Windows operating system and software applications running on a computer.

API = Application Program Interface

The main task of the Windows API is to take care of the communication between user applications and the operating system, as any software application designed for Windows uses Windows API. For example, events in Windows such as keystrokes, movement of the mouse and the insertion of a CD call on functions in Windows API that execute the event.

WriteFile	Writes data to a file.
CreateProcess	Starts a process.
SetFocus	Gives focus to an object.
EndDialog	Closes a dialog box and removes it from the memory.
CloseWindow	Removes a window from the screen.

Figure 3.1: The figure shows some examples of the functions in Windows API

Programmers who create applications for Windows do not use the functions in Windows API directly. When a programmer who develops a program for Windows wants to communicate with the operating system, this is done through calls to library functions in the development tool. The library functions will then send further calls to functions in Windows API. A function whose task is to call another function is called a Wrapper Function.

Windows API is an old library used by all versions of Windows from Windows 95 and onward. Windows operating systems have changed with each version of Windows, as has Windows API. Windows API was originally 16-bit, but Windows API has gradually become 32-bit and 64-bit. The 32-bit API has been called Win32 API, and today 32-bit and 64-bit live side-by-side. Windows API will therefore refer to both 32-bit and 64-bit versions.

About 32-bit and 64-bit architecture

32-bit and 64-bit architecture refers to the length of the memory address used by the processor. The 64-bit version of Windows, which handles large amounts of RAM (Random Access Memory), is more effective than a 32-bit system. However, much of today's software was made during the 32-bit time. To fully benefit from a 64-bit operating system, you must have software optimized for 64-bit processing.

Windows API has services used by all Windows applications, which consists of thousands of features, divided into the following categories:

- Administration and management
- Diagnosis
- Graphics and multimedia
- Network
- Security
- System services
- Windows user interface

The administration and management part of Windows API has many services for applications and systems. One of them is the installation and configuration of software applications.

The diagnosis part of the Windows API handles problems that occur during program execution, including functions for logging, processing errors, troubleshooting and more. The diagnostic part of the Windows API is attempting to fix bugs and system errors. What can happen if a program has a problem is that the Windows API manages to remedy the defect or offers to restart the program.

The Graphics and Multimedia section of the Windows API provides applications for the functionality to print graphics to the screen, printers and similar equipment. The graphics and multimedia section of the Windows API allows applications to incorporate formatted text, graphics, audio and video, and have functions to draw lines, curves and shapes, as well as use pens, images and colors. Microsoft DirectX is a collection of APIs that takes care of tasks related to multimedia on Microsoft platforms. DirectX is also a tool used in game programming.

The Network part of the Windows API has facilities that make it possible to achieve communication between applications on different computers within a network. There are several standards such as Windows Sockets, Remote Procedure Call (RPC) and Simple Network Management Protocol (SNMP).

Windows API has many services for security, and some of them include passwords at logon, the protection of shared system objects, access control, rights, monitoring, encryption and more.

System Services in Windows API provides applications access to resources on the computer such as memory, file systems, input/output devices, hardware devices, processes, threads and more.

Windows API has functions that deal with the user interface of Windows. For example, there are functions in the Windows API that create and manage windows, in addition to functions that manage controls such as menus, cursors, icons and combo boxes. Windows API also contains a number of features to manage windows, geometric shapes, text, fonts, scrollbars, dialog boxes, menus and similar GUI features.

3.2 .NET FRAMEWORK

.NET Framework is a software framework that runs on Microsoft Windows.

A *framework* in programming terms, is a collection of Application Programming Interfaces (APIs) and a shared library of code that developers can call when developing applications, so that they don't have to write the code from scratch.

Programs written for .NET Framework execute in a software environment named Common Language Runtime (CLR). CLR is an application virtual machine that provides services such as security, memory management and exception handling.

There are several advantages to having applications run inside a runtime environment. The biggest is portability. Developers can write their code using any of a number of supporting languages, including C#, C++, F#, Visual Basic, and others. That code can then be run on any hardware on which .NET is supported.

Windows Version	*.NET Framework Version*
Windows 8	4.5
Windows 8.1	4.5.1
Windows 10	4.6
Windows 10 Version 1511	4.6.1
Windows 10 Version 1607	4.6.2
Windows 10 Version 1703	4.7

Figure 3.2: The figure shows the latest versions of .NET Framework.

The .NET Framework is a part of Windows. The services of .NET Framework improves developer productivity and increases safety and reliability within Windows applications.

3.3 OBJECTS AND HANDLES

An object in the Windows operating system is a structure used to store data. An object consists of system-defined data types called attributes and functions that treat the attributes.

Objects are widely used in Windows operating system, e.g. files, windows, pictures, semaphores, processes and threads. Nonetheless, not all data in Windows operating system are objects, as only data that needs sharing, protection or to be visible to user programs is located in objects.

About Windows Objects

Objects in the Windows operating system are similar to objects used in object-oriented programming. However, objects in Windows lack some of the characteristics in object-oriented programming systems, such as inheritance and polymorphism.

The Windows operating system distinguishes between three types of objects. There are user objects, graphical objects (GDI Objects) and kernel objects. The task of user objects is to support the management of windows, while the task of graphical objects is to support graphics. Kernel objects support memory management, process execution and communication between processes.

Objects have the following important tasks in the operating system:

- To provide the names of system resources that are readable for humans.
- To share resources and data between processes.
- To protect resources from unauthorized access.
- To keep track of data that is no longer in use, and is ready for deletion.

To increase security, each object has a Security Descriptor attached. A Security Descriptor contains information about which operations are allowed to perform on the object.

In Windows NT, API functions create objects in the kernel mode. Every time a new object is created, a handle to the object is returned.

A handle is a type of pointer used in Windows. A user application does not have direct access to the data objects. Therefore, the applications use handles in order to reach and change the data in objects.

Handles are used to perform operations on objects, just as pointers and references do in object-oriented programming. To get ahold of data and functions in an object by means of a handle is faster than looking it up using the name of the object.

3.4 WINDOWS RUNS IN TWO MODES

The Windows operating system runs programs in two modes:

- User Mode
- Kernel Mode

Software applications run in user mode, whereas processes associated with the operating system runs in kernel mode.

The purpose of two modes in Windows is to increase the security of the operating system, as using two modes protects the operating system from applications that can damage or alter critical data belonging to the operating system.

The difference between kernel mode and user mode is the access that processes have to system resources. Processes running in kernel mode have access to all hardware and can perform as many tasks as possible at the computer. Processes belonging to applications running in user mode only have limited access to system resources.

3.5 PROCESSES AND THREADS

The terms program, process and thread are closely related in the theory of operating systems. Programs give rise to processes and threads, although processes and threads are quite different from programs.

Term	*Definition*
Program	A program is program code and corresponding data that resides at one or more files on a hard drive.
Process	When a program starts and is loaded into the computer memory, it creates a process.
Thread	A thread is a portion of a process code to be executed in the processor, and is the dynamic part of a process.

Figure 3.3: The figure provides a definition of a program, process and thread.

The management of processes and threads is a central task for an operating system. The operating system must start and terminate processes, allocate resources to processes, take care of the synchronization of processes, organize communication between processes and protect processes from each other.

Processes and threads are central concepts in the theory of operating systems, and we shall consider them in more detail later in the chapter on processes, threads and jobs.

3.6 SERVICES, FUNCTIONS AND ROUTINES

In connection with operating systems, we have the concepts of function, routine, subroutine and service:

1. In the C programming language, a function is a piece of code in a program called by other parts of the program.
2. A routine is defined as a piece of code in a program that is used repeatedly by calls when the program is run.
3. A subroutine can be defined as a unit with program instructions for performing a specific task.
4. A service is usually one or more routines that execute when you call them.

The meaning of function, routine and service is a matter of definition. In the following, we shall look at how these terms are used in connection with different parts of the Windows operating system.

In conjunction with the Windows operating system, the following terminology is often used:

- Windows API functions
- Native System Services
- Kernel functions
- Windows services
- Dynamic Link Libraries

Windows API functions are documented subroutines called from user applications. You therefore use the word functions in connection with library routines in the Windows API.

Native System Services in Windows are undocumented services running in kernel mode. Drivers can call these services directly, whereas applications in user mode can call them via Windows API. For example, when a user application creates a file, the function CreateFile in Windows API will call an internal system service with the name NtCreateFile. Many system services have a name that starts with Nt, in which Nt refers to the Native System Services.

Kernel Support Functions or Routines are subroutines in the Windows operating system only called from the kernel mode.

Windows Services are processes that run in the background, and perform some function. The Windows operating system has approximately 50 default services, e.g. DNS is used to find an IP address for a web address. Another example of a Windows service is a Web server that runs in the background.

It is also possible for users to add their own services. You can configure Windows services to start automatically when you restart, but you can also start Windows services by hand. Many Windows Services can be seen in the Windows Task Manager, where they often have names such as System, Local Service or Network Service.

Dynamic Link Libraries (DLLs) are subroutines used by applications in user mode, and Windows programs often use DLLs. The advantage of DLLs is that multiple processes can share them, which leads to a better utilization of the memory when the operating system ensures that only one copy of a DLL is in the memory at a time.

The opposite of DLLs are Static Libraries, in which code is copied into each application that uses them.

3.7 UNICODE

Unicode is a standard for representing characters written on a computer, for example characters used in a word processor, and are an international character set that can represent all the characters in the world.

The most widely used standard for characters has previously been ASCII (American Standard Code for Information Interchange). This character set is suited for the English language, though it only has 256 characters, which is not enough in an international world. For example, Chinese, Korean and Japanese have 10,000 commonly used characters. Consequently, Unicode was developed. Unicode has 65,536 characters, which is sufficient for all the languages in the world.

The advantage of Unicode is that it allows multilingual communication, including the representation of multiple languages in a single document. For many languages, it is difficult or impossible to do this without using Unicode.

3.8 CONTROL QUESTIONS

1. For what is API an abbreviation?
2. What is the task of the Windows API?
3. How do programmers use Windows API?
4. What is an object in Windows and why is the Windows operating system using objects?
5. What is the task of a handle in the Windows operating system?
6. Windows operating system runs in two modes. What are the names of these two modes?
7. Give a definition of the concepts of program, process and thread.
8. What is Unicode, and why do Windows use Unicode?

4 THE ARCHITECTURE OF WINDOWS

In this chapter we look at the inner structure of Windows. Windows NT is the basis for today's Windows, and we shall first look at some aims for Windows NT.

4.1 SOME AIMS FOR THE WINDOWS NT OPERATING SYSTEM

When planning Windows NT in the early 1990s, the following objectives for the operating system were set.

1. Extensibility
2. Transferability
3. Reliability and robustness
4. Compatibility
5. Performance

The operating system needed to be expandable, meaning that the code to the operating system had to be able to grow and change with the times and development.

The operating system also needed to be transferable, which means that the operating system had to work on different types of hardware architecture. The operating system should also be easily adapted to new types of hardware architecture that would come in the future.

A reliable and robust operating system has to cope with bugs in their own code. An error in an application running on the system should not be able to harm the operating system or user applications running.

When Windows NT was launched, it was compatible with earlier versions of Windows and MS-DOS. This means that applications for earlier versions of Windows and MS-DOS had to be able to run on Windows NT.

The operating system has to perform well, which means that the operating system will work as quickly as possible on different types of platforms.

4.2 ABOUT THE STRUCTURE OF WINDOWS

There are different ways to organize an operating system. One way is to create the operating system as a big application that runs on a computer such as a user application. However, this is not the way that modern operating systems were created, instead consisting of several layers, in which each layer has a specific task.

The architecture of the Windows operating system is a layered system with the two main components of user mode and kernel mode. User applications run in user mode, while processes belonging to the operating system run in kernel mode.

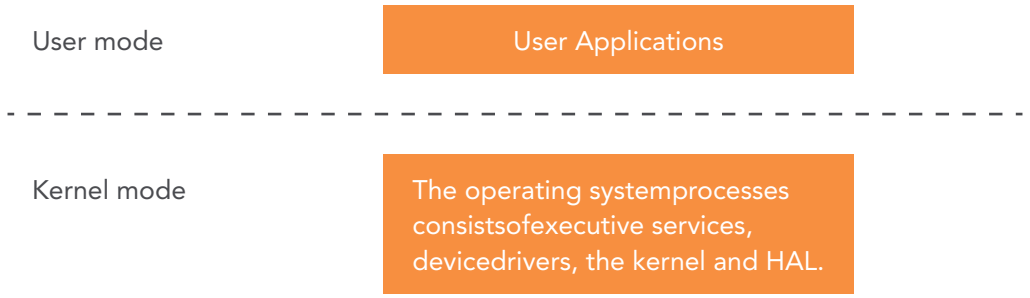

Figure 4.1: Processes are running in two modes

Central components in kernel mode are the kernel, a hardware layer (HAL), drivers and a variety of services called Executive Services. Another important component of the kernel mode is the device drivers, which manage physical devices attached to your computer, the file system, network protocols and more.

4.3 USER MODE

To protect the operating system from user applications, which can damage or critical alter data belonging to Windows operating system, applications runs in user mode. User mode also protects applications against each other.

Applications in user mode communicate with the kernel through the Windows API and access the hardware through the operating system. Applications in user mode cannot access data in other applications' memory unless through Windows API.

Processes from user applications have a lower priority than the processes belonging to the operating system.

4.4 KERNEL MODE

Processes in kernel mode have full access to the hardware and system resources. To get good security, processes in kernel mode runs in a protected area of the memory.

The key parts of kernel mode are the kernel, the executive services, the hardware layer and device drivers.

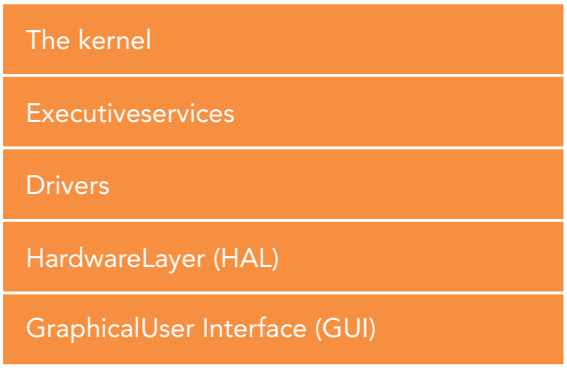

Figure 4.2: The organization of the kernel mode

The purpose of the kernel mode is to prevent applications in the user mode from being able to access critical areas of the operating system. Applications in the user mode must get help from the kernel mode to perform these kinds of operations.

4.5 WINDOWS EXECUTIVE SERVICES

Executive Services are services in the Windows operating system that help applications in the user mode to run on the computer system. Executive services include many basic services in the Windows operating system.

The operating system file Ntoskrnl.exe

The upper layer in the OS file Ntoskrnl.exe is the executive part of Windows, while the lower layer in Ntoskrnl.exe is the kernel. The executive part of Windows has the name Windows Executive.

The executive services in Windows are in kernel mode. The executive services do central services for the operating system, including the management of processes and threads, memory management and processing of input/output.

Some of the main components of the Windows Executive Services:

- Object Manager
- Configuration Manager
- Process and thread Manager
- Input/output Manager
- Installation and configuration Manager
- Memory Manager
- Cache Manager
- Power Manager

Executive services also include a variety of other features that are important to the infrastructure of the Windows operating system.

4.6 THE KERNEL

The kernel refers to the most central part of the operating system, and controls the use of the processors. The kernel manages the scheduling of threads, context switching, interrupt signals, exception signals and multiprocessor synchronization.

The kernel sits between the Hardware Layer (HAL) and the executive services. The kernel forms a bridge between user applications and data processing at the hardware level.

The most important task for the kernel is to allow applications to run and give applications access to hardware. To run a program, the kernel sets up an address space for the program, loads the program code into the memory and provides the program with a stack.

Other important tasks for the kernel are to manage your computer resources and to allow applications to use these resources. These resources are:

1. The processor – Processes get access to the processor by the kernel.
2. The computer's memory – The kernel has full access to the computer's memory and allocates memory to the processor as needed.
3. Input/output devices – The kernel receives requests from processes to perform input/output, and provides access to relevant equipment.

When Windows is starting, the kernel is also started. The kernel starts the drivers needed for the Windows operating system to start and run. The kernel runs in kernel mode, and sets up paging and virtual memory. It then starts some system processes and lets them run in user mode.

The kernel keeps track of hardware devices connected to the computer by having a list of this equipment. The kernel provides processes with access to hardware devices when needed.

4.7 HARDWARE ABSTRACTION LAYER

Hardware Abstractions are a set of routines that give applications access to hardware devices regardless of which manufacturer the equipment belongs to.

The abbreviation for the Hardware Abstraction Layer is HAL, which is a layer between the physical hardware and the software that runs on the operating system. On a PC, HAL can be seen as a driver for the motherboard that allows instructions from high-level programming language to communicate with low-level components like hardware.

A task for HAL is to get the Windows operating system to be able to work with different hardware platforms. HAL is therefore an interface to the hardware equipment in the computer operating system. Programs in Windows will not communicate directly with hardware, but communicate with hardware through HAL.

4.8 DEVICE DRIVERS

Drivers are control programs that enable communication between hardware devices and the operating system, and are necessary for Windows to communicate with hardware.

Windows includes installation drivers that support a huge number of different hardware devices. Thousands of drivers are available through a Windows Update, and there are hundreds of new drivers every month.

Device Drivers are small programs designed to help applications at a higher level to communicate with hardware. Device Drivers are running in kernel mode, and provide an interface between the input/output manager and hardware.

There are several types of device drivers. The following provides an overview of some of them:

- Drivers for hardware devices. Using the hardware layer, these drivers take care of input/output for physical equipment.
- File system drivers are Windows drivers that take care of input/output to files.
- File system filter drivers perform tasks such as encryption or writing data to more than one disk. They also do scanning to locate viruses. File system drivers for the network transfer the file system input/output to other computers at a network.
- Protocol drivers implement a networking protocol such as TCP/IP or NetBUI.
- Software drivers are kernel modules that perform operations that can only be done in kernel mode on behalf of some user mode process.

Microsoft introduced the Windows Driver Model (WDM) to allow driver developers to write device drivers that are source-code compatible across all versions of Windows. Kernel-mode drivers that follow WDM rules are called *WDM drivers,* and there are three kinds of WDM drivers: bus drivers, function drivers and filter drivers.

- A bus driver services a bus controller, adapter, bridge, or any device that has child devices. Bus drivers also detect and report other devices that are connected to the bus. Each type of bus, such as PCI, PCMCIA and USB, on a system has a bus driver.
- A function driver is the main device driver and services an individual device.
- A filter driver is used to add functionality to a device or existing driver. It is often used to fix hardware that provides incorrect information about its hardware resource requirements.

The Windows Driver Foundation (WDF) simplifies driver development by providing two frameworks: The Kernel Mode Driver Framework (KMDF) and the User Mode Driver Framework. Developers can use these frameworks to write drivers for Windows.

4.9 THE WINDOW AND GRAPHICS SYSTEM

A graphical user interface (GUI) is a type of interface that allows users to communicate with a computer in ways other than using the keyboard.

The window and graphics system implements the graphical user interfaces for Windows. This system takes care of processing windows, drawing in windows, processing controls such as menus and toolbars, receiving mouse and keyboard input and more.

4.10 SYSTEM PROCESSES

There are many system processes that always run in any Windows system, and we will look at some of these:

- Idle Process
- Session Manager
- Logon process
- Windows Subsystem
- Service control manager

Idle Process consists of one or more threads in the kernel running in the processor when there are no other threads running on the computer. In a computer system with multiple processors, there is an Idle Process for each processor.

The original reason for an Idle Process was to avoid the special situation for Process Manager of when no programs were running on the system.

In computer science, a session is an active exchange of information, also called a dialogue between two devices. Session Manager is the first process created in user mode, and is responsible for many important steps in the Windows startup. Session Manager also creates system variables, starts the kernel and user modes, creates virtual memory and starts winlogon.

The Windows Logon Process manages the login and logout of the computer system. A secure keystroke combination for starting Winlogon has been Ctrl + Alt + Delete.

The Windows Subsystem consists of several components that have key responsibilities in the operating system, including the management of the keyboard, mouse and screen.The Windows Subsystem is required for Windows to run.

The Service Control Manager is a special system process that is responsible for starting, stopping, and interacting with service processes. It is a protected process, making it difficult to tamper with.

4.11 CONTROL QUESTIONS

1. Why is the Windows operating system divided into user mode and kernel mode?
2. What is the difference between user mode and kernel mode?
3. Which layers are in the kernel mode?
4. What are the tasks for Executive Services in the Windows operating system?
5. What are the tasks for the kernel in the Windows operating system?
6. What are the tasks for the hardware layer (HAL) in the Windows operating system?
7. What are the tasks for the device drivers? Mention some types of device drivers.
8. Mention the tasks for the following system processes: Idle Process, Session Manager, Winlogon and the Windows Subsystem.

5 SYSTEM MECHANISMS

The Windows operating system has many mechanisms that are used by components in kernel mode. These mechanisms are used by the Executive services, the kernel and the device drivers.

Some of these system mechanisms are interrupts, exceptions, object manager, synchronization, global flags and wow64.

In the following, we will be looking at some of the most important system mechanisms.

5.1 INTERRUPTS

Windows is an interactive operating system that waits for events to occur. Examples of such events are key pressure, the movement of a mouse, and insertion of a CD or USB pen. When such events occur, the processor must execute them. Moreover, this must happen immediately, so the processor must interrupt what it is doing to handle the event.

Windows is using interrupts to carry out events as soon as they happen. This means that an event interrupts the thread running in the processor, so that the processor can execute the event instead.

The advantage of using interrupts is that the processor is able to work undisturbed most of the time. The processor only gets an interrupt when it is necessary to perform an event or a service.

An interrupt is an asynchronous event, which means that the event can occur at any time, and is independent of what happens in the processor.

All hardware devices can use interrupts to receive services from the operating system. Not only hardware, but also the software system can send interrupts.

Interrupt signals from the hardware mostly come from input/output devices that need to make the processor aware that it must perform a service. However, almost all types of peripheral devices may use interrupts as the primary method to get the operating system to perform actions.

An interrupt service routine is a routine that hardware calls on in response to an interrupt signal. Windows has a table for interrupt signals, which locates the interrupt service, and is designed to handle a specific interrupt signal.

5.2 EXCEPTIONS

Exceptions are mechanisms that allow applications to handle errors or unexpected events during program execution.

Exceptions are similar to interrupts. However, unlike interrupts, which can take place at any time, exceptions can only arise from program execution.

Exceptions are sent when an error occurs during program execution. Some examples of conditions that can cause exceptions are division by zero, insufficient memory to perform an operation, stack faults, page faults in memory and floating-point errors.

Programmers can use exceptions in the applications code to handle errors or situations, where something does not go according to the plan. For example, what happens if an application wants to open a file and the file is not on the hard drive? If opening the file causes an exception, the program can switch to a code written to process the error that occurred.

```
try
{
      StreamReader sr = new StreamReader(filename);
      String line;

      // Reads and displays lines from the file.
      while ((line = sr.ReadLine()) != null)
            Console.WriteLine(" " + line);

}
catch (Exception e)
{
      // Gives a message about what went wrong.
      Console.WriteLine("Couldn't read the file: ");
      Console.WriteLine(e.Message);
}
```

Figure 5.1: The program code above shows the use of an exception in a C# program. If the program cannot open or read the file, an exception will occur and in this case a message will be printed out on the screen.

Windows uses a system called structured exception handling, which gives applications the opportunity to gain control of the situation when an error occurs. The program can then try to fix the error itself with a code written in case an error should occur.

When an exception happens, the system will generally take care of (save) the current code to be executed, and program execution switches over to a subroutine called an Exception Handler. Depending on the situation, the interrupted code can start again when the Exception Handler is finished.

5.3 OBJECT MANAGER

Object Manager is a part of the executive services in Windows, and its task is to manage Windows resources. All resources are treated as objects, with some examples of these being processes, threads, files, semaphores, drivers and events.

One can say that an object is a data structure in the kernel mode available for virtual memory. Each object has a head and a body. The head contains information (Properties) used by the object manager, while the body contains code and data.

Objects can be either kernel objects or executive objects. Moreover, applications running in user mode can use executive objects through executive services.

Only the kernel use kernel objects. The kernel objects represent resources such as devices or services such as synchronization.

The Object Manager takes care of the following tasks:

1. Create and return handles to objects.
2. Making multiple handles to an object if necessary.
3. To check if a process has the right to use an object.
4. Keeping track of which quotas belong to which resources.
5. Delete objects and close the handles to objects.

In total, the Object Manager takes care of the lifecycle of objects from creation to deletion. The Object Manager is very important because it takes care of the management of system resources and data structures.

The operating system uses objects to regulate access to the system for two reasons.

1. The use of objects ensures that Microsoft can update the system's functionality. When new versions of the operating system are made, the properties of objects can be changed without causing major consequences for the system.
2. The use of objects increases the security of the Windows operating system. Each object has a list that specifies what actions are possible on the object.

5.4 32-BIT APPLICATIONS ON A 64-BIT OPERATING SYSTEM

The WoW64 subsystem is a layer in the Windows operating system that allows 32-bit applications to run on a 64-bit system without having to make changes to the software:

- WoW64 stands for a Windows 32-bit on a Windows 64-bit.

WoW64 consists of DLLs that allows 32-bit applications to run on 64-bit versions of Windows. WoW64 is included with all 64-bit versions of Windows.

5.5 CONTROL QUESTIONS

1. What is the advantage of using interrupts?
2. How does the Windows operating system use interrupts?
3. In what contexts does the Windows operating system use exceptions?
4. How can programmers use exceptions?
5. What is the role of the Object Manager?
6. What is the function of WoW64?

6 MECHANISMS FOR DATA MANAGEMENT

Windows has many system settings that must be handled and taken care of, and are stored in a database called the registry. The Configuration Manager takes care of- and manages the registry.

6.1 THE REGISTRY

The registry is a hierarchical database that contains information about the various settings used by the operating system. The registry contains system settings used at the Windows startup. An example of this is user settings such as the colors and background on a desktop.

The registry plays an important role in setting up and controlling the Windows operating system. It keeps track of system settings, user settings and what hardware and applications are on the computer.

The registry contains information about the applications installed on your PC, in addition to information about the processor, memory, drivers, network and similar aspects.

Windows has quite a bit of important information in the registry, including:

- Settings that determine the layout of Windows.
- What drivers that belong to different equipment.
- What software to start at boot time.

Windows retrieves data in the registry under the following conditions:

1. When starting the computer (Boot Process).
2. When loading the kernel (Kernel Boot Process).
3. When a user logs onto the system.
4. At the startup of applications.

The registry is organized into separate volumes called Hives, with each Hive set in its own file in the director:

C: \ Windows \ system32 \ config

The Configuration Manager takes care of- and manages the registry.

6.2 CONFIGURATION MANAGER

The Configuration Manager is an important component of the Windows executive services, which take care of- and manage the registry.

We have seen that data is stored in registry files called hives, and the Configuration Manager reads data from these files.

The configuration to a computer will change over time. The Configuration Manager updates the registry as programs and other components of the operating system change settings.

At startup, the Configuration Manager provides the ability for computer to receive this new configuration.

6.3 CONTROL QUESTIONS

1. What is the role of the registry in the Windows operating system?
2. What kind of data is stored in the registry?
3. In what situations do Windows retrieve data from the registry?
4. What is the task of the Configuration Manager?

7 PROCESSES, THREADS AND JOBS

A process, thread and job are key concepts associated with programs running on a computer. We will therefore begin with a definition of these terms.

Process

When we start a program and it is loaded into the computer memory we have a process, which is a program that runs on the computer.

Thread

A thread is a unit of code to execute in the processor. A thread belongs to a process and is called the dynamic part of a process.

Fiber

A fiber is a kind of thread, and is often called a lightweight tread. A fiber is scheduled in a different way than ordinary threads.

Job

Windows can group processes that cooperate, with such a group of processes called a job.

7.1 PROCESSES

A computer program is one or more files on a hard drive containing code and program resources. When a program is started and is loaded into a computer's memory, we have a process.

A process is thus a program running on a computer. A program can give rise to multiple processes on a computer if someone starts the program several times at the same time.

A process is work to do on a computer. A processor runs a process by executing the process's code. On a computer, there are often many processes, but only one or a few processors. Consequently, if there are more processes than processors, they must execute in the processor or the processors in turn.

A process consists of two parts:

- The static part
- The dynamic part

The static part of a process is resources on your computer that the process uses. The static part of a process can consist of a piece of memory that belongs to the process, a directory, an open file and other related resources.

The dynamic part of a process consists of one or several threads. The dynamic part of a process is a program code that runs on your computer, i.e. as a set of instructions executed one after another in the processor.

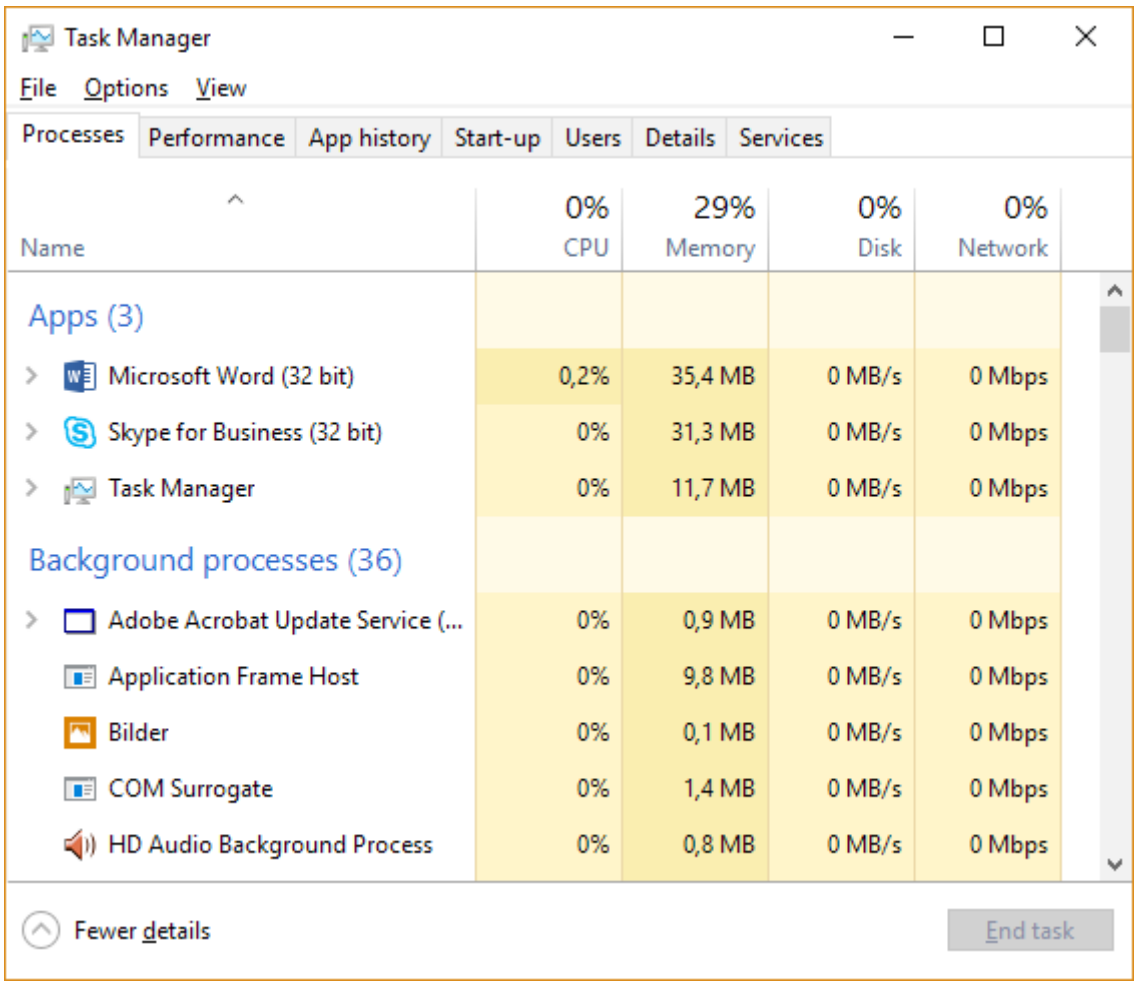

Figure 7.1: Task Manager in Windows shows the processes running on your computer

In the Windows operating system, a process is defined in the following way:

- A process is a container in the memory with the resources required to run a program on the computer.

A Windows process consists of:

1. An address space consisting of a set of virtual addresses.
2. An executable program.
3. At least one line of code to execute in the processor.
4. A list of handles to various system resources.
5. A unique process ID used to identify the process.
6. A security structure that regulates access to the process.

For each process that is running, the operating system has a structure called a Process Descriptor, which includes the following information:

1. The name of the process.
2. The start time for the process.
3. The status of the process.
4. A list of threads related to the process.
5. A description of addresses in the memory.
6. A list of resources that belong to the process.

The Process Manager takes care of processes, threads and the resources that belong to the processes. The Process Manager therefore controls the activities of the process and the resources used by the process.

7.2 THREADS

A thread comprises some code to execute in the processor. A process can create one or more threads, and if a process has multiple threads, the threads will run independently and compete for time in the processor.

A process that has only one thread is a single-threaded process, while a process with multiple threads is called a multithreaded process. The advantage of multithreaded processes is that they can perform different tasks simultaneously.

Why use multiple threads?

One reason to use multiple threads in an application is that several activities are going on simultaneously. One of these activities can sometimes stop, for example, waiting for input from the user. It may then be fine if other activities can continue to work independently, which they can do, if they are separate threads.

Using multiple threads is particularly beneficial on computers with multiple processors, as the threads run in parallel for increased speed.

Since threads are competing for processor time, each thread has a status. The status is either active or inactive, meaning that a thread is either active in the processor or on a standby status.

State	*Description*
Blocked	The thread is in a wait state.
Ready	The thread is ready to be loaded into the processor.
Active	The thread is executed in the processor.

Figure 7.2: The three possible states for a thread

In Windows, a thread is a process object to which the operating system is allocating processor time. The process itself is a primary thread, which can launch other threads. The process has a list with the threads that belongs to the process.

A thread has a Thread Descriptor in the computer's memory with information about the thread. A Thread Descriptor is a data structure that the operating system uses to obtain information that it needs to process a thread, and has the following information about the thread:

1. A description of the thread.
2. The thread status.
3. The start time for the thread.
4. A reference to the Process Descriptor.
5. A list of threads associated with the thread.
6. A reference to the thread's resources.

All threads in a process have common access to the resources belonging to the process. Most of the resources used by a thread are allocated to the process, though resources can be allocated directly to a thread.

There are usually many threads running simultaneously on a computer, and the operating system must therefore allocate time for them in the processors. A processor is running thread after thread, and thus shifts constantly between running threads.

Context Switching is the procedure to remove a thread from the processor and to load in a new thread. When Context Switching has removed a thread from the processor, the thread is on standby status. All information about the thread is then stored so that the thread can continue to run in exactly the same way the next time it comes into the processor.

In general, several threads compete to get into the processor. Scheduling determines the order in which the threads will enter the processor, so the main task of a scheduler is to allocate time in the processor for threads in the best possible way.

One way to organize scheduling is to give threads different priorities, as threads with a high priority have easier access to the processor than those with a low priority.

7.3 FIBERS

Fibers are quite new in the Windows operating system, and were introduced in Windows Vista. A fiber is code to execute in the processor like a thread. A fiber is a sub-thread that belongs to a conventional thread, and fibers are often called lightweight threads.

A fiber is a unit of execution that must be manually scheduled by the application. Fibers allow an application to schedule its own threads of execution rather than rely on the priority based scheduling mechanisms built into Windows operating system. In terms of scheduling, they are invisible to the kernel because they are implemented in user mode.

Fibers require less support from the operating system than threads, and in contrast to threads, fibers may be using cooperative multitasking. Cooperative multitasking means that a thread is in the processor until it voluntarily relinquishes the place.

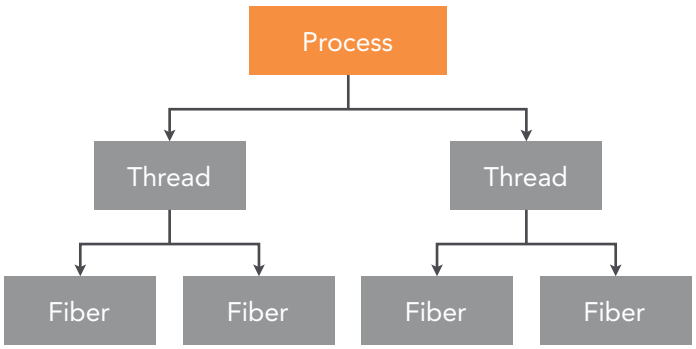

Figure 7.3: The relationship between processes, threads and fibers

An advantage of fibers is that it is easy to change from one fiber to another in a processor. To switch between fibers only requires storing and the reading of some registers, while to switch between threads requires them to go out of- or into the kernel.

Using fibers is usually not a good idea. This because they are invisible to the kernel. Additionally, fibers cannot run concurrently on more than one processor, and are limited to cooperating multitasking only. The best is usually to use threads and to let Windows kernel handle the scheduling.

7.4 JOBS

It is possible to group processes so that they can work together. In Windows, such a group of processes is called a job. A process can only be in one job at a time.

To group processes in a job is only applicable when the processes need to cooperate.

In the Windows operating system a job is an object. The function of a job object is to handle groups of processes as a unit. A job object is a kernel object with a name, and which possesses a high security.

7.5 THE PROCESS MANAGER

A Process Manager manages processes and threads, and takes care of the following tasks:

- Starting and ending processes and threads.
- Provide parameters and characteristics of processes and threads.
- Allocation of resources to processes and threads.
- Implementation of input and output.
- Implementation of addresses in the memory.

The Process Manager does not do operations on processes and threads such as scheduling and synchronization, instead being taken care of by the kernel.

7.6 MULTITASKING

Previously, a PC just had one processor, which could only run one thread at a time. This is sufficient if there is just one program running on your computer, as it was for the first PCs. However, when graphical operating systems were developed, it became possible to run multiple programs in different windows. It was then necessary for computers to be able to run multiple programs simultaneously. As a result, an operating system like Windows thus has a need for multitasking.

Windows supports multitasking, which means that two or more threads can run simultaneously on one processor while taking turns in the processor.

Multitasking allows a user to run multiple applications simultaneously and to switch between applications. For instance, while simultaneously working with a word processor and a spreadsheet, you may transfer the data from the spreadsheet to the editor by cutting and pasting.

A multitasking operating system gives processor time to the threads that need it. Multitasking is done by Context Switching. In using Context Switching, a thread runs in the processor until the operating system interrupts it or the thread voluntarily leaves the processor.

We distinguish between two types of multitasking:

- Cooperative multitasking.
- Preemptive multitasking.

In cooperative multitasking, programs share CPU time between them. A thread holds the processor until it voluntarily gives it away.

Preemptive multitasking ensures that the various processes only run for a short time in the processor. A thread can only be in the processor for a time interval called a quantum. When a quantum is finished, the operating will automatically do a context switch.

7.7 COMMUNICATION BETWEEN PROCESSES

Inter-process communication is when a thread in a process interacts with a thread in another process. This applies when:

1. The processes are on the same computer.
2. The processes are on different computers.

Communication between processes on the same computer is quite straightforward. The operating system can copy data from the memory to the sending thread to the memory of the receiving thread.

Communication between processes in different computers is as follows. The operating system on the computer that sends will copy the data from its memory to a communication unit. The communication unit will then send the data to the receiving computer.

Windows operating system have several mechanisms to support communication between processes:

Clipboard

A Clipboard is widely used to transfer data from one application to another. When a user copies or cuts data from an application, the data can be transferred to another application by pasting it.

File Mapping

File Mapping is a way for a process to manage the content of a file as if it were in the memory of the process. The process can use a pointer to retrieve data from the file and modify the contents of the file. File Mapping is a simple way for two or more processes to share data, but the method requires synchronization.

Pipes

Pipes are a simple model for process communication. One thread writes data to a buffer, so that another thread can read the data from the buffer. We distinguish between two types of pipes: Anonymous Pipes and Named Pipes.

Moreover, Anonymous Pipes can be used to transfer data between threads in the same process.

Named Pipes can transfer data from one application to another application on a computer, as well as transfer data from one computer to another computer over a network. Named Pipes has a name that will allow other processes to identify them. In Windows, named pipes are based on a client-server communication similar to that used by Sockets.

7.8 CONTROL QUESTIONS

1. What is the static part, and what is the dynamic part of a process?
2. What is the definition of a process associated with the Windows operating system?
3. Give a definition of a thread.
4. Under what conditions can a program benefit from using multiple threads?
5. What is Context Switching?
6. What is the difference between a thread and a fiber?
7. How is scheduling done for threads?
8. How is scheduling done for fibers?
9. When may it be appropriate to group processes in a job?
10. What is multitasking?
11. What is the difference between cooperative and preemptive multitasking?
12. What is the difference between Anonymous Pipes and Named Pipes?

8 SYNCHRONIZATION OF THREADS

There are often multiple threads simultaneously running on a computer. It may then be necessary to synchronize the threads to avoid collisions.

There is a need for the synchronization of threads in two situations:

1. Threads that cooperate.
2. Threads competing for a resource.

8.1 THREADS COOPERATING

Threads can cooperate in various situations. For example, a thread can be responsible to perform a task for another thread.

Example of thread cooperation

Two threads working towards a common data structure, such as a list, is an example of cooperation between threads. One thread can have the task to create and hang items on the list, while the other thread can remove items from the list and use them. The second thread is then dependent on the first thread.

If all the threads in a computer system were running independently without having anything to do with other threads, it would not be difficult to allocate time to them in the processor. What makes processing difficult is that the threads can cooperate and rely on each other. The operating system must therefore be able to manage cooperation between threads.

8.2 THREADS COMPETING

A competition between threads occurs when two threads want to use the same resource at the same time. An example of this is two threads that write to the screen simultaneously, which could lead to collisions and messy prints.

Two threads writing to the screen at the same time may provide collisions. For example:

- Thread 1 writes Hello.
- Thread 2 writes Hi.

A result on the screen could be HeHillo. However, in practice, the printing will be a lot messier. Collisions can cause the positions of the letters to crash, and the letters will then be written out at random locations on the screen.

Many resources can only use one thread at a time. If two threads try to use such a resource at the same time there will be a collision. Hence, to avoid collisions, an operating system must be able to manage competition between threads.

8.3 CRITICAL REGION

The term *critical region* is important in connection to the synchronization of threads. A critical region is a place where collisions may occur. In traffic, an intersection can be a critical region because cars can easily collide there. Therefore, traffic lights, roundabouts and traffic rules control intersections.

A critical region in conjunction with an operating system is a piece of a program code that attempts to obtain a resource that cannot be shared. If different threads use the same resource at the same time, this can lead to collisions with the result that an application does not function normally.

A file is an example of a resource that cannot be shared. If two threads write to the same file at the same time, the contents of the file will become an unreadable mix of characters.

Another example of a resource that cannot be shared is a printer. If two processes are printing their documents at the same printer at the same time, the printout will probably not be readable.

In addition, variables are resources that cannot be shared. If two threads attempt to change the value of a shared variable at the same time, collisions may lead to the loss of one of the updates. The variable will then not have the correct value afterwards.

Example of resources simultaneously used by multiple threads are hardware devices, memory blocks, and files and shared variables. A thread that will use such a resource should have exclusive rights to the resource when it is used.

8.4 MUTUAL EXCLUSION

Mutual Exclusion is an important concept in connection to synchronization. That threads can get mutual exclusion to a resource is necessary for a computer system to function properly.

We have seen that if two threads access a resource at the same time, it may lead to collisions. It is therefore important that only one thread at a time can access a resource that cannot be shared. If only one thread at a point of time has access to a resource so that other threads cannot access, the thread has Mutual Exclusion to the resource.

Having Mutual Exclusion to a resource is required when the resource cannot be shared, because sharing the resource will lead to unpredictable results. Consequently, operating systems must use different mechanisms to ensure that threads have a mutual exclusion.

In the following, we shall look at some commonly used mechanisms used for synchronizing threads.

8.5 SEMAPHORE

One way to control access to a resource is to use a semaphore. A semaphore may be described as an integer, and can have the value 0, 1, 2…, *n,* where *n* is a number that indicates how many threads are allowed to access the resource at the same time.

For instance, two threads that are simultaneously writing to the screen can use a semaphore to avoid collisions. The semaphore can either have the value zero or the value one.

Value semaphore	*State*	*Consequence*
0	The screen is in use.	No more threads can access.
1	The screen is not in use.	One thread can access.

Figure 8.1: A semaphore can provide mutual exclusion to a resource

Before a thread writes to the screen, it checks the semaphore. If the value of the semaphore is one, the value is set to zero and the thread has mutual exclusion to the screen. When the thread is finished using the screen the value of the semaphore is set to one again, so that other threads can then access the screen:

```
MySemaphore.WaitOne();
// Now, only this thread has access to the screen.

Console.Write("Message from the thread: {0}", message);

MySemaphore.Release();
// Releases the semaphore for other treads to use the screen.
```

Figure 8.2: The above code shows the use of a semaphore in a C# program.Printing to the screen using Write() is a critical region. Therefore, a semaphore is used to ensure that only one thread is allowed to use the screen at any given point in time.

A semaphore can have more values than only zero and one. It may sometimes be possible that multiple threads can simultaneously access a resource. You can then use a semaphore with several values.

For example, a semaphore with a numeric value of 0, 1, and 2 will be able to allow two threads simultaneously. The semaphore can start with a value of 2, and each thread that then enters the semaphore decreases the value by 1. When the value is 0, then no more threads can access the resource.

Semaphores can regulate access to resources, but there are some disadvantages to using semaphores that you must bear in mind:

1. Threads are not forced to use semaphores. Some threads may not use a semaphore when it is necessary.
2. If a thread with a low priority that is starved gains access to a semaphore, this will lock the system. This is because other threads with a higher priority are unable to access the semaphore.
3. The incorrect use of a semaphore can lead to deadlock.

Many programming languages support semaphores. For this reason, semaphores are still widely used by programmers.

8.6 MUTEX

Mutex is an abbreviation for Mutual Exclusion. A Mutex is a synchronization mechanism made for cases when you want the mutual exclusion of a resource, in addition to being a kind of semaphore that only can have the values zero and one.

Semaphores that have more values than zero and one do not provide the mutual exclusion of a resource because multiple threads can simultaneously access the resource. For a semaphore to provide mutual exclusion, it must only allow the values zero and one.

A difference between a Mutex and a semaphore is that semaphores are dependent on the operating system to function. A Mutex may be implemented by more specialized and faster procedures. It is therefore advisable for programmers to use a Mutex instead of a semaphore when the objective is to obtain mutual exclusion.

8.7 SOME SYNCHRONIZATION MECHANISMS IN WINDOWS

There are many synchronization mechanisms other than semaphore and Mutex. In the following, we will review some synchronization mechanisms used in the Windows operating system:

- Keyed Events
- Spinlocks
- Fast Mutex and Guarded Mutex
- Push locks

Keyed Events

Event objects are a type of synchronization mechanism used in cases in which threads cooperate. An event object is used when a thread must wait until an event has occurred before it can perform some task. This event may be that another thread has finished a job.

An example of the use of Events

Given two threads and a file, one thread writes to the file while another thread reads the file. This gives an example of when the two threads must wait for each other. Thread 2 cannot read the file before Thread 1 has written to it. Moreover, Thread 1 cannot write to the file again before Thread 2 has read it.
An event object can be used to synchronize reading and writing to the file.
When Thread 1 has written to the file, it updates the event object, and before reading the file, Thread 2 then can check the event object to be certain whether it is written to the file or not. When Thread 2 has read the file, it updates the event object.
Thread 1 and Thread 2 can constantly keep abreast of the read/write status of the file using the event object.

Windows uses a synchronization mechanism called Keyed Events, which is an object that allows a thread to specify a key for what event it is waiting for. The thread will wait until another thread signals that an event with the same key has occurred.

Spinlocks

Spinlock is a locking mechanism used by the kernel in a Windows operating system to achieve the mutual exclusion of a resource. Before a thread in the kernel can access a critical resource, it must ask a spinlock for access. If the spinlock is not free, the thread will keep trying to access the spinlock until it becomes available.

The thread is waiting in a loop (spinning) until the spinlock becomes available. The thread will then be in a wait state (not active), while waiting for the spinlock to become available. Once a thread has access to a spinlock, it will have the mutual exclusion of the resource until it lets go of it again. To use a spinlock is therefore only effective if threads only have the spinlock for a short period.

Queued Spinlocks are also used. A Queued Spinlock has a queue, and when a thread attempts to access a queued spinlock that is busy, the thread is placed in a queue belonging to the spinlock. When the spinlock becomes available, the thread that is first the in queue is the next to access the critical resource.

Fast Mutex and Guarded Mutex

The Windows operating system uses two types of Mutex, which are called Fast Mutex and Guarded Mutex.

Fast Mutex provides better performance than a common Mutex. The reason for this is that it can bypass normal process planning, which provides Fast Mutex with a particularly good performance in environments with multiple processors. Fast Mutex is widely used in communication with drivers.

Guarded Mutex is used by Windows Server 2003 and later versions of Windows, and is used for the same tasks as Fast Mutex, but is even faster.

Push locks

A push lock is a synchronization mechanism that was first introduced in Windows Server 2003. Push locks are used instead of spinlocks to protect data structures in the kernel, with one advantage of push locks being that they use little space.

Access to a push lock can be both exclusive and shared, and can be shared if data is to be read. If a thread shall be able to change the content of the data, access to the data must be exclusive. If a thread has exclusive access to a push lock, other threads trying to access the push lock will be put on a waiting list.

Push locks is only used by Windows operating systems, but there are also some drivers who use push locks.

8.8 CONTROL QUESTIONS

1. Explain why there is a need for the synchronizing of threads.
2. What is a critical region in conjunction to the theory of synchronization?
3. Provide some examples of resources that only can be accessed by one thread at a time.
4. What is mutual exclusion in connection with the synchronization of threads?
5. Explain the behavior of a semaphore.
6. What is the difference between a semaphore and a Mutex?
7. In what situations is the synchronization mechanism Keyed Events used?
8. Explain the behavior of a spinlock.

9 PROCESS PLANNING ON ONE PROCESSOR

There are usually more threads running on a computer than there are processors. It is therefore necessary to organize how threads shall use the processor or the processors.

Planning how threads should use one or several processors does not differ from other planning, such as planning the use of classrooms at a school or planning a traffic flow using traffic rules.

In this section, we will look at process planning for multiple threads on one processor, and we will return to process planning on multiple processors in the next chapter.

Processor scheduling is the task of managing the sharing of a processor between different groups of threads. The scheduler that is a part of the operating system is taking care of process planning.

There are usually several threads on a computer waiting to execute in the processor. When a thread has finished processing and a new thread is to be loaded, the process manager must decide which thread is to be next.

A policy for process planning determines which thread is the next to be loaded into the processor, and must determine two things:

- When the time has come for a thread to leave the processor.
- Which thread is the next to be loaded into the processor?

There are four reasons why a thread leaves the processor:

1. The thread is finished, for example at the program's end.
2. The thread needs a resource to continue, but the resource is currently not available.
3. The thread decides to voluntarily leave the processor.
4. The thread must leave the processor, because the time interval in the processor is finished.

Process planning can have a huge impact on computer performance since it determines when a thread is to be loaded into the processor. Ignoring a thread and never loading it into the processor will lead to the starvation of the tread.

There are many ways to do scheduling for threads, as how to organize scheduling is a classical research problem for operating systems. Modern algorithms for process planning tend to use priority, i.e. the different threads have different priorities, and threads of higher priority enter the processor more easily.

The importance of process planning

Process planning is much more important on a server in a network than for a single PC, as there are often many processes on a server competing to get into the processor.
Process planning is not so important on individual PCs. A person using a PC often works with only one application at a time, such as a word processor or a web browser. Today's PCs are so fast that there is no problem to run a few processes.

9.1 VOLUNTARY AND INVOLUNTARY SHARING OF THE PROCESSOR

Threads can share the use of a processor in two ways, through:

- Voluntary sharing.
- Involuntary sharing.

Voluntary sharing of the processor means that threads leave the processor by themselves, as the voluntary sharing of the processor is the same as cooperative multitasking. One problem with using the method of voluntarily sharing the processor is that there may be threads that are not programmed to cooperate.

Involuntary sharing of the processor means that there is an interruption system that can interrupt a thread running in the processor. This system employs a timer that interrupts the thread when a certain time interval has passed.

The Process Manager determines which threads get to enter the processor, as well determining how long the threads will be in the processor. How the Process Manager is organized has a major impact on the computer's speed and performance, as threads that readily enter the processor will have a high performance. Conversely, a thread will have a low performance if neglected, and rarely enters the processor.

Process scheduling algorithms designed for modern operating systems use priority to determine which processes get to access the processor. Each thread has a priority that indicates how important the thread is.

One can distinguish between two types of algorithms for process planning:

- Non-preemptive algorithms.
- Preemptive algorithms.

Non-preemptive algorithms allow a process that has come into the processor to run until it is finished or until it leaves the processor by itself.

Preemptive algorithms allow processes to run for only a specific time interval, and threads are allowed to be executed in the processor for just a specific time interval.

9.2 NON-PREEMPTIVE PROCESS PLANNING

Some theoretical ways an operating system using a non-preemptive process planning may work:

1. The job that comes first is first performed.
2. The smallest job first.
3. Priority process scheduling.
4. Deadline process planning.

The job that comes first is first performed strategy allows threads to come into the processor as they are started. One way to organize this is to have a list with the threads that are ready to go into the processor. The threads are then placed back on the list as they come. Whoever is first on the list then comes first into the processor, but since this strategy does not work well in practice it is rarely used.

The smallest job first strategy is that the thread that is fastest to the finish comes first into the processor. This strategy reduces the waiting time because the small threads are running away fast. The disadvantage of this strategy is that it can lead to the starvation of large threads if there are too many small threads.

Priority process scheduling means that threads are chosen to run in the processor by priority, as a process with a high priority will more easily enter the processor than a process with a low priority. The disadvantage of this method is that it can lead to the starvation of threads with low priority.

In real-time systems, it is often necessary that some threads have to run in the processor in order to keep a deadline. System performance depends on such deadlines, which is called *deadline process planning*.

9.3 PREEMPTIVE PROCESS PLANNING

Some ways that an operating system using a preemptive process planning can work:

1. Round Robin.
2. Queues with different priorities.
3. Foreground and background processes.

Of all the scheduling algorithms, Round Robin is the most widely used. The threads are running in the processor in turn, with each thread processing a specific time interval called a Quantum. All threads are treated equally without priority, as the idea behind Round Robin is that all threads will get equal time in the processor.

Queues with different priorities comprise a system in which there is a queue for each priority. Threads with the same priority are hung on the same queue, so queues with a different priority are thus an extension of the priority system. There are many different strategies for giving threads on all queues time in the processor.

Foreground and background processes are one way to organize threads. Threads in the foreground are running on the processor to perform tasks for a user, whereas threads in the background get into the processor when there is an opportunity, i.e. when there are no active threads in the foreground.

9.4 SCHEDULING FOR VARIOUS OPERATING SYSTEMS

How to organize process planning depends of the kind of operating system being used. For example, real-time processes in a real-time operating system must respond immediately. Real-time processes must therefore have a much higher priority than usual processes.

What scheduling algorithm is best depends on the type of operating system in use, and we will look at three different systems:

- Batch System
- Interactive System
- Real-time System

Batch systems perform a task described in a file. The first computers used batch systems, and batch systems are still used today in the business world.

In batch systems, no user is waiting for a response, so there is no need for preemptive algorithms. A batch process is running in the processor until it is finished.

In an interactive system, a user interacts with a computer's operating system. This gives a need for preemptive algorithms, which use context switching. This replacement of threads makes it possible to work with several applications at once, and ensures that your computer does not halt if an error occurs in a process.

Real-time systems often have sophisticated algorithms for process planning, involving processes that must respond immediately to have a high priority. Moreover, these processes must always be ready to respond to external events within the system.

The following gives the aim of process planning for these three types of operating systems:

1. The aim for the process scheduling of algorithms for batch systems is to perform as many jobs as possible in the shortest possible time, i.e. to keep the processor active at all times.
2. The aim for the process scheduling of algorithms for interactive systems is to be able to react to events as quickly as possible. It must be able to satisfy user's expectations of the system.
3. The aim for the process scheduling of algorithms for real-time systems is to be able to react to events immediately. The system must be predictable, and things have to happen at the right time.

The aim for the process scheduling of algorithms for all systems is to provide all the processes access to the processor, i.e. to exploit the system in the best possible way.

9.5 CONTEXT SWITCHING

Context Switching is the process to remove a thread from a processor and to load in a new thread. The most common causes of a context switching are:

- The time for a thread to execute in the processor is finished.
- A thread with a higher priority has become ready to run.
- A thread running has to wait for a resource. When a thread has to wait, it is taken out of the processor.

A thread in the processor uses multiple registers with data and information about the thread. When context switching happens, data in these registers are saved, so that they can be loaded into the processor the next time the thread execution continues.

The Process Manager has a queue for each priority with threads that are next to execute in the processor.

The following occurs when a thread leaves the processor:

1. The data for the thread leaving is stored.
2. The thread leaving is put at the end of the queue in its priority class.
3. A new thread in the queue with the highest priority is looked up.
4. The new thread is removed from the queue and is loaded into the processor.

9.6 QUANTUM

Preemptive algorithms ensure that each thread gets a moment in a processor called a quantum. Once a quantum is finished, the system interrupts the thread running and another thread with the same priority is loaded into the processor.

However, the fact that a thread must leave the processor when a quantum is over, is not always unconditional. If there is not another thread with the same priority, Windows will allow a thread to run a quantum more.

How long a quantum is varies with the type of operating system.

In client versions of Windows, a thread is about two clock intervals in processor, whereas in Windows Server, a thread is about 12 clock intervals in the processor.

The length of a clock interval varies with the type of hardware platform. For an x86 single processor, a clock interval is approximately 10 milliseconds. For x86 and x64 multi-processors, a clock interval is approximately 15 milliseconds.

9.7 ABOUT SCHEDULING IN WINDOWS

MS-DOS and early versions of Windows did not have multitasking, and therefore did not have a process manager. Windows 95, 98 and Millennium used non-preemptive process planning; hence, processes would then run in the processor until they were finished.

Today, Windows is a preemptive multitasking operating system, meaning that the operating system can carry out a context switching to satisfy a process scheduling policy. Preemptive process planning makes it possible that a process with a high priority takes a processor from a process that is active in the processor.

Operating systems using preemptive multitasking guarantees that each thread gets some time in the processor, thereby making the preemptive multitasking reliable.

The Process Manager manages multitasking by determining which of the competing threads will be the next to execute in the processor, and the Process Manager in Windows uses priority to determine this.

The priority of each thread is determined by the following:

- The priority level of the thread.
- The priority class of the process.

Threads are scheduled by priority, with each thread having a priority that varies between 0 and 31. The priorities from 0 to 15 are normal priorities, while priorities from 16 to 31 are real-time priorities with respect to threads that must react quickly.

Processes also have priority, and a process belongs to one of the following priority classes:

IDLE_PRIORITY_CLASS
BELOW_NORMAL_PRIORITY_CLASS
NORMAL_PRIORITY_CLASS
ABOVE_NORMAL_PRIORITY_CLASS
HIGH_PRIORITY_CLASS
REALTIME_PRIORITY_CLASS

A process will normally belong to NORMAL_PRIORITY_CLASS. Screensavers are an example of threads that can have an IDLE_PRIORITY_CLASS.

A HIGH_PRIORITY_CLASS and REALTIME_PRIORITY_CLASS should be used with caution, as using them can cause other threads to not get enough time in the processor.

9.8 CONTROL QUESTIONS

1. What is processor scheduling?
2. What is the difference between non-preemptive algorithms and preemptive algorithms for process planning?
3. How does Round Robin work?
4. What is the difference between the batch system, the interactive system and the real-time system?
5. Give some reasons why the operating system performs context switching.
6. How long is a Quantum?
7. How is process planning done in the Windows operating system?

10 PROCESS PLANNING ON MULTIPLE PROCESSORS

Previously, all PCs had just one processor, called the Central Processing Unit (CPU). In recent years, it has been common to have multiple processors in a personal computer, which run in parallel and increase the speed of the system.

About computer speed

In the history of the PC, the speed of computers has steadily increased every year. The usual way to speed up has been to increase the computer's clock speed. However, there is a limit to how fast signals can be transmitted in a computer system. Moreover, the faster the signal transmit, the more heat it develops, thus leading to overheating. The new trend for increasing a computer's speed is to therefore to use multiple processors.

Multiple processors provide particularly good performance when the number of threads is the same as the number of processors. Performance also improves when there are more threads than processors, but not to the same extent as when each thread has a processor to them self.

10.1 MULTI-CORE PROCESSORS

The core is the part of a processor that reads and executes instructions. Previously, processors only had one core, while today's new PCs have one processor with multiple cores.

A multi-core processor is a processing system with two or more cores that has an integrated circuit with two or more individual processors, which in this context are called cores. A dual-core processor has two cores, while a quad core processor has four cores. A multi-core processor implements multiprocessing in a single physical package.

Cores connect to a multi-core device either tightly or loosely, e.g. cores can share or not share cache and memory.

Tightly coupled multiprocessors also have multiple processors that connect at the bus level.

Loosely coupled multiprocessors form a cluster that consists of one or two multiple individual processors connected by a high-speed communication system.

It is also possible to make clusters of computers to acquire more processor power. A computer cluster consists of loosely interconnected computers that work together, so that they can be considered a single system.

10.2 THE ORGANIZATION OF MULTIPLE PROCESSORS

Multiple processors in computers make new demands of operating systems. Multiprocessor operating systems are mostly like single processor operating systems, but must have enhanced features in terms of process scheduling, synchronization and the management of resources.

We will now look at some ways to organize the use of multiple processors.

Each processor has its own operating system

One way to organize the use of multiple processors is that each processor has its own operating system, which helps to avoid excessive collisions. Each processor does not have its own entire operating system, but does have a copy of the data structures to the operating system, while sharing the operating system code. The collaboration of different processors via a shared memory, is hence an advantage.

This model, in which each processor has its own operating system, was used in the past, though it is rare today because it is a poor use of system resources.

Master-Slave model

Another way to organize operating systems with multiple processors is the Master-Slave model. In this model, the operating system is running in one of the processors (Master), whereas the user applications are running in the other processors (Slaves). This method provides an easy way to organize the use of multiple processors in a computer.

Figure 10.1: The Master-Slave model with four processors

The disadvantage of the Master-Slave model is that all slave processors must communicate with the operating system in the master processor. The master processor therefore becomes a bottleneck, and all processors are dependent on the master processor. The consequence of this is that processes in the slave processors must often wait for the master processor.

Equivalent processors

Another way to organize multiprocessor operating systems is to have the operating system into memory and allow any processor to execute threads. All processors are equal and can run both operating system threads and threads that belongs to applications. Threads are just loaded into the processor that is first available.

This model is called symmetric multiprocessing or SMP shortened. Using SMP, there are no overload of one processor like the Master-Slave model.

10.3 SYMMETRIC MULTIPROCESSING

SMP (Symmetric Multi-Processing) is the computer architecture that modern operating systems use. The SMP processors all share one operating system and a shared memory. SMP allows each processor to run any thread. SMP can easily move tasks between processors so that tasks evenly balance between the processors.

Figure 10.2: With SMP, each processor run threads from both user applicationsand threads belonging to the operating system

A problem with SMP is that there may be collisions if two processors try to access the same resource at the same time. One way to get out of this problem is to use a Mutex so that two processors cannot use the same resource at the same time. This works, but the problem is an inefficient system because the processors must wait for each other, and then of course some of the intent with multiple processors falls away.

However, a Mutex can improve a system that regulates access to the operating system since many parts of the operating system are independent of each other. For example, one processor can process scheduling, while another processor processes the file system. A Mutex controls each critical region in the operating system so that only one processor can use it at a time.

Most modern operating systems use SMP, but this needs to be done with care to avoid deadlocks.

10.4 MULTIPLE PROCESSORS AND SYNCHRONIZATION

Multiple processors working simultaneously increase the need for synchronization. The processors use a shared memory, and with threads working in parallel this can easily lead to collisions. There is therefore a need for Mutex's to synchronize critical regions; however, the disadvantage of using a Mutex is a delay in time.

It is possible to treat a thread that must wait in two ways: either the thread is spinning or the system performs a context switch.

Spin	Spinning means that a thread is waiting for a Mutex until it becomes free. With spinning, the thread is repeatedly trying to gain access to a Mutex, as long as the Mutex is locked.
Switch	A thread that is waiting (spinning) for a Mutex, can be taken out of the processor to release room to another thread.

Figure 10.3 Definition of spin and switch

Spinning is a waste of time. However, context switching takes time to perform. What is best will depend on how long a time the thread is spinning. Spinning will pay off if the thread just has to wait for a short time, whereas context switching will only pay off if the thread must wait for a while.

In practice, it may be beneficial to use both methods. The system keeps track of the last spin time, and uses this to decide whether to use spinning or context switching.

10.5 PROCESS SCHEDULING ON MULTIPLE PROCESSORS

When there is only one processor, it is only necessary to do a scheduling of the threads. If there are several processors, the system must also schedule the use of processors. This gives scheduling in two dimensions, which is much more complicated than scheduling for only one processor.

Sometimes threads are cooperating, which makes process scheduling more difficult when there are multiple processors. This means that some of the threads running on the system are independent of the others, while other threads cooperate with each other. In that case, it would be beneficial to coordinate threads that are cooperating.

We will look at two ways to organize scheduling on multiple processors: time-sharing and space-sharing.

Time-sharing	This organizes process planning as for threads working independently. Depending on priority, the threads hang on one or more lists, and the threads get into a processor when it becomes available. This organizing is simple and effective.
Space-sharing	Here, process planning is organized as for threads that cooperate. The system waits until each thread can have a processor each. Then the threads start simultaneously and can cooperate. An advantage of space-sharing is that context switching is not necessary for the threads to cooperate.

Figure 10.4: Definition of time-sharing and space-sharing

A disadvantage of space-sharing is that sometimes a processor is blocked and nothing happens. A system that combines time-sharing and space-sharing may then be desirable, with Gang scheduling being one way to do this.

The Gang scheduling process takes place as follows:

1. Groups of threads that cooperate are treated as a unit called a gang.
2. All members of the gang run simultaneously on different processors as with timesharing.
3. All gang members start and end their time slices together.

Gang scheduling combines time-sharing and space-sharing to ensure a high overall system throughput and short response times for interactive tasks.

The advantage of threads running simultaneously is that cooperation takes place immediately. A thread will easily be able to communicate with another thread if both run at the same time.

10.6 MULTIPROCESSING IN WINDOWS

Multitasking is a technique in which multiple threads share a single processor. However, it is an advantage that a computer has multiple processors, because then threads can run parallel.

Windows is a symmetric multiprocessing system (SMP). SMP run threads on any available processor, which ensures that processors and resources are efficiently utilized. Windows attempts to schedule threads on the processor that is most optimal for the tread.

It has been an objective for the Windows operating system to work well on multiprocessor computer systems. Originally, Windows supported up to 32 processors, but some versions support even more processors. 64-bit versions of Windows can support up to 64 processors.

Windows supports multi-core processors, as well as two modern types of multiprocessor systems:

- Hyper threading
- Non-uniform Memory Access (NUMA)

Hyper threading

Hyper threading is a technology introduced by Intel that supports multiple logical processors on one physical processor. For each physical processor core, the operating system creates two virtual processors and distributes tasks between them when possible.

Hyper threading works by duplicating parts of the processor without duplicating the device that performs the instructions, and is used to improve parallel actions performed by processors on PCs. This system requires that the operating system support multiple processors.

NUMA systems

In Non-uniform Memory Access (NUMA) systems, processors are grouped into smaller units called nodes. Each node has its own processor and memory, and connects through a cache-coherent coordinated bus.

By the use of NUMA, a processor has faster access to its own memory than a shared memory. The advantage of a separate memory for each processor is to avoid that several processors attempt to work on the same memory, which causes delays.

NUMA makes memory accesses faster, which gives modern computers better performance.

10.7 CONTROL QUESTIONS

1. Why have multiple processors become common in PCs?
2. What are the disadvantages of the Master-Slave model?
3. Describe how symmetrical multi-processing (SMP) works.
4. Why is scheduling more difficult when a computer has multiple processors?
5. Describe process scheduling by time-sharing and room-sharing.
6. Describe how Gang Scheduling works.
7. How does Hyper threading work?
8. What are the benefits of NUMA systems?

11 DEADLOCKS

When multiple threads are competing for the same resources it can cause something called a deadlock, which refers to a lock that cannot be opened.

A thread in a deadlock stops running because it is waiting to access a resource. As long as the thread cannot access the resource, it will continue to wait. The thread can therefore wait forever if it does not obtain access to the resource.

For a deadlock to occur, there must be two or more threads involved. Two threads with a status deadlock are in a situation where each thread is waiting for a resource that the other thread occupies. Both threads will be in a blocked state.

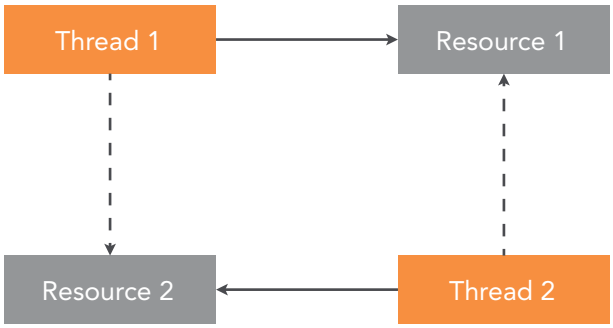

Figure 11.1: A deadlock with two threads and two resources. Thread 1 has access to Resource 1, and tries to access Resource 2. Thread 2 has access to Resource 2, and tries to access Resource 1.

An example of a deadlock is when two threads have each opened a file for writing, and both threads want to open the other thread's file. The figure below illustrates the situation:

	File 1	File 2
Thread A	Has File 1 open.	Trying to open File 2.
Thread B	Trying to open File 1.	Has File 2 open.

Figure 11.2: Example of deadlock with two threads and two files

Both threads attempt to open the other thread's file, but a thread cannot access a file that is open for writing. Both threads will therefore wait forever.

Sometimes there are more than two threads and two resources involved in the same deadlock. For example, thread 1 is waiting for a resource that thread 2 has, thread 2 is waiting for a resource that thread 3 has and thread 3 is waiting for a resource that thread 1 has.

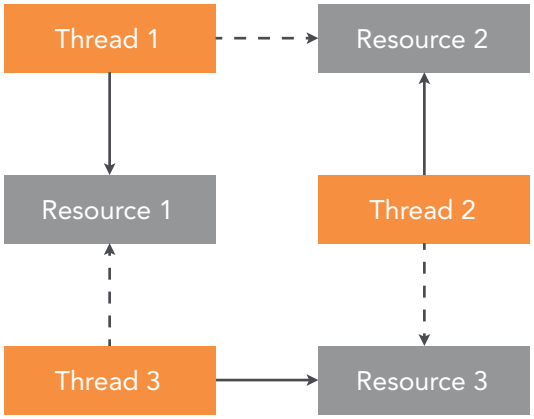

Figure 11.3: Deadlock with three threads and three resources. The figure illustrates why deadlocks are called circular holding.

The following gives a definition of deadlock:

- Some processes are in a deadlock if each process is waiting for an event that only one of the other processes can cause.

Deadlock is a potential problem in any operating system. Deadlock can occur in a variety of ways, and there will always be two or more threads involved in a deadlock.

11.1 EXAMPLES OF DEADLOCKS

In a computer system, many resources can only be used by one thread at a time. These are resources such as printers, CD players, files and similar items. A deadlock can occur when two threads are both trying to get hold of a resource that the other thread controls.

An example of a deadlock

The following is a practical example of a deadlock. Two processes both want to scan an image and to write to a DVD. Both processes therefore have use of the same scanner and DVD.

Process 1 first accesses the scanner and Process 2 first accesses the DVD. Process 1 then attempts to access the DVD, and Process 2 tries to access the scanner at the same time. This results in the situation in the figure below:

	Scanner	DVD
Process 1	Has access to the scanner.	Attempting to access the DVD.
Process 2	Attempting to access the scanner.	Has access to the DVD.

Figure 11.4: Example of deadlock with a scanner and a DVD

Both processes will stop running and wait for each other, so there is a deadlock.

Deadlock in a Database

A deadlock may occur in many different situations. The following gives an example, in which two processes are using a database:

1. Process A closes some data in the database.
2. Process B closes some other data in the database.

Now, if both processes try to use the data that the other process has closed at the same time, a deadlock occurs.

Deadlocks in a network

Deadlocks can occur in networks with several computers involved. Often, printers, scanners and similar devices are resources in the network. A deadlock can then occur if multiple computers are simultaneously attempting to get hold of shared resources in the network.

Communication deadlock

Communication deadlock is a type of deadlock that does not have to do with resources. Suppose that Process A sends a request to Process B, and then Process A blocks until it gets a reply. If the response from Process B becomes lost, this will lead to a situation similar to a deadlock.

Starvation is similar to a deadlock

Starvation is an issue that is closely related to a deadlock. Threads, which do not enter the processor, are in a locked situation similar to a deadlock.

11.2 CONDITIONS FOR A DEADLOCK TO OCCUR

For a deadlock to occur, we must have the following conditions:

1. The mutual exclusion condition
2. The hold and wait condition
3. Non-preemptive condition
4. The circular wait condition

The mutual exclusion condition means that if a thread has access to a resource, no other threads can access the resource.

The hold and wait condition describes the situation of a deadlock. The threads are holding a resource, and are waiting for access to another resource.

The non-preemptive condition means that the operating system will not intervene and take a resource from a thread. The resource will therefore not be released unless the thread willingly let go of it.

A process will keep a resource until it lets go of it. The circular wait condition is when process P1 holds resource R1 and process P2 holds resource R2, while process P2 is trying to obtain resource R1 and process P1 is trying to obtain resource R2. This situation may also involve more than two threads.

11.3 HOW THE OPERATING SYSTEM CAN MANAGE DEADLOCKS

The operating system can deal with deadlocks in four ways:

1. Ignore deadlocks.
2. Try to find deadlocks and correct it.
3. Try to avoid that deadlocks occur when allocating resources.
4. Try to avoid deadlocks by avoiding the conditions for them.

1 Ignore deadlocks

One way to deal with deadlocks is to ignore them, which may be acceptable if deadlocks are a rare phenomenon. If a computer stops working once a year because of a deadlock, it will have little impact if the computer can easily restart.

2 Finding deadlocks and correct it

The operating system can try to find deadlocks and correct it, as there are techniques developed to detect deadlocks. Operating systems that do this are usually not trying to prevent deadlocks.

If the operating system has detected a deadlock, there are different strategies to remove it:

1. Restore through that the operating system intervenes.
2. Restore through reset.
3. Recovery by ending processes.

Restore through that the operating system intervenes means that the operating system takes a resource from a process and gives it to another process. However, it is often difficult or impossible to correct a deadlock in this way.

Restore through reset means that a process in a deadlock is set back in time so that it can begin the allocation of resources again. To achieve the reset of processes, something called checkpoints are used. The checkpoints of a process mean that the status of the process is stored in a file at specified times, which means that the process can start again from a checkpoint.

Recovery by ending processes means that the operating system terminates one or more processes to eliminate a deadlock, which may give other processes an opportunity to proceed.

3 Try to avoid that deadlocks occur when resources are allocated

Instead of trying to find and correct deadlocks, operating systems may attempt to avoid deadlocks. The operating system can try to avoid a deadlock from occurring when allocating resources.

4 Try to avoid deadlocks by avoiding the conditions for them

The operating system can also try to avoid deadlocks by avoiding the four conditions for deadlocks:

1. The mutual exclusion condition
2. The hold and wait condition
3. The non-preemptive condition
4. The circular wait condition

The mutual exclusion condition implies that only one process at a time can access a resource. However, this condition is difficult to use to avoid deadlock, because mutual exclusion is necessary for many resources such as printers, files and similar items.

Hold and wait condition means that for a deadlock to occur, processes will hold a resource while it attempts to access another resource. One way to avoid this is to require that a process give away the resources it has, before it gains access to a new resource.

A non-preemptive condition means that the operating system will not take a resource from a process. To take a resource from a process can have adverse consequences for the process, e.g. to take a printer from a process that is in the middle of a printing would be drastic. However, it is possible to avoid deadlocks in connection with the printers if the process writes the document to a file on a hard drive. The process can then continue and the printer can print the file later.

A circular wait condition means that for a deadlock to occur, a process must attempt to simultaneously access multiple resources. This means that if a process only tries to access one resource at a time, deadlocks will not occur.

11.4 TOOLS TO FIND DEADLOCKS

Driver Verifier is a tool in Windows, designed to find bugs in drivers. Hardware manufacturers can use Driver Verifier to test whether drivers are working correctly under Windows, and are not making illegal function calls or creating system damage.

Driver Verifier can also check for possible deadlocks by choosing the option known as Deadlock Detection, which can explore the use of Spinlocks and Mutex's, as well as looking for patterns that might indicate deadlock. If Driver Verifier detects a deadlock, it stops the system and gives an indication of which driver is causing the deadlock.

11.5 CONTROL QUESTIONS

1. Give an example of a deadlock.
2. What conditions must be present for a deadlock to occur?
3. What is circular holding in connection to deadlocks?
4. Why do deadlocks resemble starvation?
5. Give some ways an operating system can deal with a deadlock.
6. How can the operating system attempt to avoid deadlocks from occurring?

12 MEMORY MANAGEMENT

Managing memory concerns in terms of how to allocate memory for programs and how to free memory that is no longer needed. How this is done will affect computer performance, which makes memory management an important task in a computer system.

The following provides an overview of some of the problems with memory management:

1. There must be space for several programs in the memory at the same time.
2. There may not be room for all programs in the memory at the same time.
3. Programs will have different addresses in the memory at different runs.

Models for treating memory must consider these issues. We will now be looking at some ways to organize the management of memory.

12.1 SWAPPING

If there is not space for several processes in the memory at once, it is possible to use swapping. Swapping means that the processes take turns in disposing the memory.

A process is loaded into the memory and runs for a while. The process is then removed from the memory and another process is loaded into the memory and runs for a while. In this way, all processes take turns in disposing the memory.

Swapping is an old-fashioned method that has previously been used. Modern operating systems use virtual memory, which allows applications to run without loading whole programs into the memory.

12.2 SOME MEMORY MODELS

Most modern memory models fall into three categories:

1. A Flat Memory Model
2. A Paged Memory Model
3. A Segmented Memory Model

A Flat Memory Model uses linear indexing for memory addressing, as the indexes start at 0 and go up to a maximum. The processor uses indexes to directly access data stored in the memory, with a flat model providing a simple organization of the memory. It needs little resources for memory management, and the system acquires fast access to data.

A Paged Memory Model is more complicated than the Flat model, and gives a little slower access to data. By paging memory, the contents are moved to a hard disk so that the memory is released to other processes or the operating system itself. Data is divided into equally sized blocks called pages. The advantage of paging is that it is not necessary to keep all programs in the memory at the same time, and that the system always work with units of the same size.

The paged memory model is suitable for multitasking, in which threads are constantly swapped in the processor. Paging is an important part of the technique used for virtual memory.

The Segmented Memory Model is similar to the Paged Memory Model, but the sides differ in size. This makes the segmented memory model more flexible and efficient than paging. By segmenting, programs are divided into independent segments addresses. Each segment has its own address space that goes from zero to a maximum. The segment length can be anything between zero and the maximum. When a program is loaded into the memory, the operating system makes a segment table that contains the starting address of each segment.

The disadvantage with the Segmented Memory Model is that it is a complex system, which is difficult to program and can easily result in failure.

12.3 VIRTUAL MEMORY

Although there is a lot of memory on a computer, it is a possible situation that there is not room for all processes to run simultaneously in the memory. The operating system put some processes or parts of processes on a hard drive, so it is then necessary to manage processes scattered on various storage media. This is usually done by a technique called virtual memory.

Virtual memory is a technique which gives applications the impression that your computer has a unified and coherent memory, even if the memory can be physically located on different media. The advantage of the virtual memory architecture is that a part of a process can reside on your hard drive if there is not enough memory space.

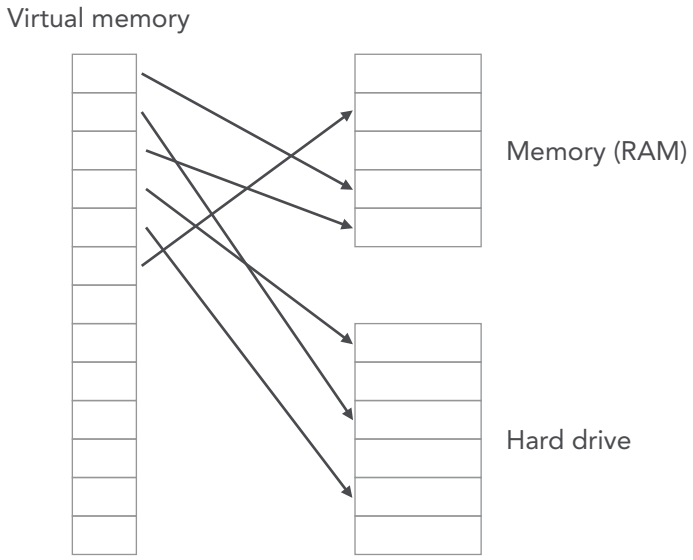

Figure 12.1: Virtual Memory points to physical memory that can be on various storage media

Almost all the implementation of virtual memory shares the virtual address space into pages. When a process is running in the processor, the processor reads the code page-by-page in the same way as humans read a book.

Operating systems use the technique of virtual memory to make it easier to use the physical memory, which makes it easier for large applications to run on computers. All modern operating systems use the techniques of virtual memory.

12.4 MEMORY MANAGEMENT IN WINDOWS

Memory management in Windows uses the virtual memory architecture. In Windows, each process has a virtual address space, which is divided into pages.

Windows has a virtual Memory Manager that uses paging, so that only the active parts of programs are loaded into the memory when they run. The processors on which Windows runs support two page sizes: small and large.

Windows supports Demand-Paged Virtual Memory. This means that the operating system only loads pages into memory that a running process requests.

A Cache Manager keeps track of virtual addresses used to map files into the memory. This improves the input/output performance for many applications because read operations can be done without using the hard drive.

In today's Windows, Win32 and Win64 coexist. Win32 has a 32-bit memory structure that uses pointers with 32-bit memory addresses, whereas Win64 has the opportunity for a much larger virtual address space because Win64 uses 64-bit pointers.

12.5 THE MEMORY MANAGER

The Memory Manager in Windows takes care of the memory management. It allocates space for programs in memory when needed, and it removes programs from memory again when they finish.

The Memory Manager is a part of Windows Executive and is also the largest component in the executive. The Memory Manager is allocating, deallocating, and managing virtual memory most of which are exposed through Windows API or kernel mode device driver interfaces.

Memory Manager implements virtual memory, perform tasks on the memory and gives support for large memories. Memory Manager also creates services such as keeping track of files that are part of the memory and gives support to the Cache Manager.

The Memory Manager has system services that allocate and free virtual memory, shares memory between processes and manages pages. Memory Manager also has many services that allocate and free physical memory.

Programmers can use most of the services in the Memory Manager through the Windows API.

12.6 THE HEAP MANAGER

Dynamic memory allocation is a method used to allocate memory to processes running on a computer. The heap is a portion of the memory used for dynamic memory allocation. Programmers can use the heap to store variables and objects. Variables and objects created in the heap will be there until deleted by the process or by the Garbage Collector.

Windows divides the memory into heaps. When each thread has a free heap, there will be no competition for the memory between threads, which increases system performance.

Each process has at least one free heap. Once the process starts, it creates a heap called the Default Process Heap, which exists until the process is completed. Processes can also create additional private free heaps using a Windows API function called HeapCreate.

The Heap Manager optimizes the performance of memory usage for allocations in the heap. Heaps are Windows objects and processes, which thus have access to the free heap by means of a handle.

12.7 STACKS

When a thread is running, it must have access to a temporary storage where it can store local variables, have function parameters and return addresses for function calls. A stack is a portion of the memory used for this.

A stack is a last-in, first-out data structure. A new element is always added at the top of a stack, and when removing an item from a stack, the top item is always removed. Items at the bottom of a stack will therefore be there until all the items over it are removed.

The Memory Manager in Windows uses two stacks for each thread, a user stack and a kernel stack. When a thread is created, it will automatically get a user stack and a kernel stack. User stacks generally has a size of 1MB, while kernel stacks are often much less, with a common size being 12 KB.

12.8 CONTROL QUESTIONS

1. What makes memory management difficult?
2. What is swapping?
3. Describe how the Paged Memory Model works.
4. What is the difference between the Paged Memory Model and the Segmented Memory Model?
5. Describe the technique of virtual memory.
6. How is memory management done in Windows?
7. What is the use of the free heap?
8. What is the task of the Heap Manager?
9. What is a stack?

13 CACHING IN WINDOWS

The purpose of caching is to make input/output run more efficiently, which is done by storing data in a cache. A cache is a piece of memory for the temporary storage of data that is expected to be used several times.

Processors use a cache to speed up the computer system. It is faster to retrieve data from a small cache near the processor than to retrieve data from a large memory. When a processor will retrieve data from the memory, it first checks whether the data is in the cache. If there is no data in the cache, it is retrieved from the memory and then added to the cache for future use.

The cache is often used in conjunction with reading hard disks. To retrieve a file from a cache is much faster than reading the file from a hard disk, so files are therefore often stored in a cache. Files that are read into a memory from a hard drive are taken care of in a cache so that the files can be retrieved quickly from the cache instead of reading them from the hard drive.

Internet browsers use caching to work faster, as it saves time to have web pages that are much read in a cache at an Internet network provider. Web pages can then be retrieved from a cache instead of being loaded down via the Internet from a distant part of the world.

13.1 THE CACHE MANAGER

The Cache Manager in Windows takes care of the organization of caching. The Cache Manager consists of a set of functions in kernel mode, as well as threads that cooperate with the Memory Manager.

An important task for the Cache Manager is to ensure that processes that are trying to obtain data get the latest version of the data, and not older versions.

The Cache Manager provides a high-speed intelligent mechanism to reduce input/output for disk drives and improve computer system performance.

13.2 CONTROL QUESTIONS

1. What is the purpose of caching?
2. Describe how caching takes place.
3. In what connections is caching used?

14 INPUT AND OUTPUT

A computer is connected to devices such as a screen, keyboard, mouse, hard disk, printer, CD/DVD, etc. An important task for an operating system is to communicate with the input/output devices connected to the computer.

14.1 PROBLEMS WITH INPUT AND OUTPUT

Input and output have been a problem in operating systems. One reason for this is the large difference in speeds within a computer system.

In the processor and the memory, data processes very quickly, though by comparison communication with peripheral devices is very slow. To write to the screen or to a file is a slow process. A user typing on a keyboard can wait several seconds before he types a character, while for the processor, it is not a good use of time to wait for input/output devices.

Type of equipment	*Transfer speed*
Keyboard	10 bytes/second
Mouse	100 bytes/second
SAS disk	129 Mb/second
SSD disk	241 Mb/second
USB 3.1	1250 Mb/second

Figure 14.1: The table provides an overview of the speed of some input/output devices

Another challenge with input/output is that hardware connecting to a computer is from different manufacturers. For example, there are many types of screens, and these require separate drivers. It would hence be cumbersome if programmers had to create code for each type of screen driver that might be using the program.

The operating system takes care of communication with hardware

It makes programming much easier when programmers can communicate with the operating system, rather than directly with the hardware. For example, in his code a programmer can write an output to the screen as Write ("A Message").
The operating system takes care of the printing to the screen, so the programmer does not have to think about communication with the various drivers that may be for screens from different manufacturers.

The operating system takes care of the communication with the screen, mouse, printers and similar equipment. Applications can therefore work on different computers with different hardware devices connected.

14.2 BLOCK DEVICES AND CHARACTER DEVICES

There are two categories for input/output devices:

- Block Devices
- Character Devices

Block Devices store data in blocks, each of which has an address. The typical size of a block has generally been between 512 bytes and 32768 bytes. Hard drives, CD-ROM and USB drives are examples of block devices.

Character Devices are another type of input/output devices than Block Devices. Character devices transmit and receive data as characters. Examples of Character Devices are the keyboard, the mouse, printers and network connections. Devices that are not discs are usually Character Devices.

14.3 DATA STREAMS

An application must open a connection to an input/output device before using it, with a stream being the term used for such a connection.

When a new process starts, it automatically creates three Data Streams:

1. Standard input from the keyboard
2. Standard output to the screen
3. Standard error to the screen

When a stream opens, it can stay open as long as needed. However, the operating system must keep track of which Data Streams are open, and thus has a data structure to keep track of streams.

14.4 TREATMENT OF ERRORS IN INPUT AND OUTPUT

The operating system takes care of errors that can occur when communicating with input/output devices. The operating system should be able to:

1. Detect errors.
2. Correct mistakes.

Many errors can occur when using input/output devices. The operating system should be able to detect whether a process tries to write to a file opened for reading, and also recognize corrupt data.

Sometimes the operating system successfully corrects errors that occur. If the operating system is able to correct a mistake, it will not normally notify the program where the error occurred.

14.5 SYNCHRONOUS AND ASYNCHRONOUS INPUT/OUTPUT

Input and output are slow processes, which means that the system must wait for input/output operations. How long the system must wait varies, from a few milliseconds to save a file to a hard drive to up to several minutes of waiting for a character from the keyboard.

Instead of waiting for input/output operations, it may be better for a thread to do something else. However, a thread often has to perform an input/output operation before it can continue. For instance, if a thread uses ReadLine() to read a line of data from the screen, the thread cannot proceed until the line is read if the thread requires the data from the line in the further program execution.

Output goes somewhat better. If we have written Write ("A Message") in a thread, the thread can proceed without waiting for the print to the screen to happen.

There are two ways for a thread to deal with input and output:

1. Synchronous I/O
2. Asynchronous I/O

Through the use of synchronous I/O, a process will wait to continue until an input/output operation is complete, as the thread goes to sleep while waiting. Another name for synchronous I/O is wait I/O.

Through the use of asynchronous I/O, the process continues to run without waiting until an input/output operation is performed. Another term for asynchronous I/O is no wait I/O.

Asynchronous I/O requires synchronization in the code of the thread. It is possible to use a semaphore, which causes the thread to wait until it gets a signal from the driver. Using a semaphore with asynchronous I/O provides the same effect as synchronous I/O. However, an advantage of asynchronous I/O is that the thread can execute some other code before the thread starts to wait for the input/output operation to finish.

14.6 USE OF A BUFFER

When doing an output, it may be helpful to use a buffer. For example, if a thread wants to write to a printer and the printer is busy, the thread can write the output to a buffer instead. The thread can then continue, and the printer can print the contents of the buffer later.

The advantage of using a buffer is to save time. Once a thread has written to a buffer, it can continue executing other codes. The disadvantage of using a buffer is that it requires space on a hard drive.

When a thread is writing to a file, it may be necessary to close the access to other processes. Locks are mechanisms used to close files.

14.7 POLLING

We will now take a look at how the operating system communicates with hardware. What happens is that signals from peripheral devices such as keyboards transfer to the computer's memory.

One way to treat the data from the keyboard is to put it in a register. The processor can then regularly check the register and update applications with this data. This method is called polling.

Using a register to take care of keystrokes has the disadvantage that it does not synchronize reading and writing to the register, as writing to the register can go faster than reading or vice versa. Two presses of the keyboard before the processor has read the register can overwrite a keystroke. Alternatively, the processor may read the same keystroke twice because there have not been more keystrokes since the last check.

To help solve this problem, two registers are used:

1. A register that takes care of the keystroke.
2. A register that indicates whether the processor has read the register or not.

In the polling method, registers are regularly controlled. However, this is cumbersome since there are many registers, and each peripheral has registers.

The polling method is not good because it lacks synchronization, and the processor may possibly check at the wrong time. Either there is nothing in the registers when the processor checks or they are overfull. It is therefore better to use interrupt signals.

14.8 INTERRUPTS

The processor should not have to waste time checking registers such as when using the polling method. Interrupts are a better method than polling since interrupt signals allow peripheral devices to notify the processor every time there is something new in a register.

With an interrupt system, the processor can keep doing what it does without worrying about peripheral devices.

The peripheral device notifies the processor when there are data to be processed. When the processor gets an interrupt signal, it ends what it is doing and reads the register.

14.9 DIRECT ACCESS TO MEMORY

Interrupt signals with data registers work for key presses and similar, although this system is not so good for reading from a file when the amount of data read into a memory is large. It is a burden for the processor to handle at lot of data like a file transfer, so a system called Direct Memory Access (DMA) is used instead.

Like polling, the DMA method uses a status register and an interrupt system. However, DMA does not use any data register, as the data transfers directly from the file into the memory.

DMA allows certain hardware subsystems to directly access memory independently of the CPU. A usual input/output transfer makes the processor busy all the time during the input/output, and the processor will not be able to perform other operations. Using DMA, the processor will initiate input/output transfer and do other tasks while the transfer takes place, in addition to receiving an interrupt signal when the transfer is complete.

DMA is an important feature of modern computer systems because it makes it possible for devices to transfer data without too much burdening of the processor.

14.10 ABOUT DRIVERS FOR INPUT/OUTPUT DEVICES

Each physical device attached to a computer needs software specifically designed for it and its tasks. A driver is software that makes it possible for the operating system to communicate with hardware, and is either a part of the operating system or a small program designed to communicate with the operating system.

When an application needs to use a driver, it calls a function that belongs to the operating system's library. The operating system then calls that driver's code. When the peripheral device is ready, it sends an interrupt signal to the processor.

The operating system calls a driver using a Device Switch, which is an array. Each driver has an index in this array that contains a collection of pointers that point to methods used to use the driver.

Different operating systems manage drivers in two ways:

1. The driver is always in the memory, so that it is sleeping when not in use.
2. The driver is loaded into the memory only when necessary.

A data structure called an IORB is important for a driver.

IORB = Input Output Request Block

This data structure contains parameters for input/output driver, and the information in an IORB can be:

- A flag indicating whether to read, print or similar.
- Where data is located in memory.
- The Number of bytes to transfer.
- The peripheral unit number.

To make sure that the drivers for hardware work well with the rest of the Windows operating system, Microsoft has defined a model called a Windows Driver Model (WMD) that drivers are expected to work with. Microsoft also has a Driver Verifier that examines many of the functions of drivers to make sure that they follow the requirements of WDM.

There are hundreds of thousands of hardware drivers for a Windows operating system, which provides a lot of code. Application errors in the driver code means that the system stops, and That Windows will then show the Blue Screen of Death. The normal program execution then stops.

14.11 INPUT AND OUTPUT IN WINDOWS

A Windows operating system provides an interface to handle a variety of input/output services and drivers. A Windows input/output system consists of many components that together treat hardware devices and provide a user interface to hardware devices for applications.

Some key components of a Windows input/output system are the Input/output Manager, device drivers, the Installation and Configuration Manager, the Power Manager and the registry.

A device driver provides an interface between the operating system and a hardware device, and receives commands via the Input/output Manager.

The Installation and Configuration Manager detects changes in the layout of the hardware, and load and removes drivers. The Power Manager tries to reduce the power consumption of hardware equipment not in use. The Input/output Manager also provides support for manipulating the input/output objects in the kernel.

The registry serves as a database that stores application settings and descriptions of hardware devices connected to the computer. The registry also has information at the startup and configuration of drivers.

14.12 THE INPUT/OUTPUT MANAGER

The Input/output Manager in Windows is the central component of the input/output system. It forms a connection between software and hardware that provides an infrastructure that supports hardware drivers.

The Input/output Manager takes care of system hardware drivers, and does a variety of services related to the equipment attached.

The Input/output Manager takes care of the following tasks:

- The implementation of drivers.
- The configuration of drivers.
- Access to hardware devices.
- Operations on hardware equipment.

14.13 THE PLUG AND PLAY MANAGER

The Plug and Play Manager in Windows takes care of connecting new devices to a computer, as well as the removal of equipment from the computer. If a new hardware device does not have a driver, the Plug and Play Manager acquires a driver for it.

The Input/output Manager includes the installation and configuration of the new hardware connected (Plug and Play). The Plug and Play Manager starts when a new hardware component is detected on the computer, and then attempts to find a driver for this hardware component.

The mission of the Plug and Play Manager is to make sure that Windows can recognize and adapt to different hardware devices, with the advantage being that users do not have to think about installing the drivers themselves. When new equipment connects to a computer, the Plug and Play Manager automatically makes a configuration and adds drivers.

Some features and functions of the Plug and Play Manager:

1. It automatically detects the installed hardware devices.
2. It allocates resources to the hardware equipment.
3. It finds the correct drivers for the hardware devices.
4. It also supports network devices such as printers.

14.14 THE POWER MANAGER

The Power Manager in Windows manages the system's power usage. Previously, reduced electricity use would turn off the monitor and the hard disk. Now, newer methods reduce the power consumption of the computer's components when not in use.

Windows supports a method for shutdown called sleep mode (hibernation), which copies all physical memory to the hard disk. Power consumption is then reduced to a minimum.

An alternative to Hibernation is Standby Mode, which also reduces power consumption to a minimum for the entire system.

There are so many computers in the world that a reduced power consumption of computers has great economic importance. The power consumption of computers has been so large that a reduction could reduce the number of nuclear power plants in the world.

14.15 CONTROL QUESTIONS

1. What makes input/output to a problem in a computer system?
2. What is the difference between block devices and character devices?
3. What is a stream? Give an example of a stream.
4. What is the difference between synchronous I/O and asynchronous I/O?
5. What are the advantages and disadvantages of using a buffer?
6. Describe the polling method.
7. Why are interrupts better than the polling method?
8. Describe the DMA (Direct Memory Access) method.
9. What is a Device Switch?
10. What are the tasks for the Input/output Manager in Windows?
11. What are the tasks for the Plug and Play Manager in Windows?
12. Describe some ways the Power Manager can save power on a computer.

15 FILE SYSTEMS IN WINDOWS

Before you can start using a new hard drive, you must format it to use a file system.

A file system stores and organizes the files on a hard disk, so it is desirable that it should be easy to find and retrieve files. File systems are therefore made for this purpose.

To take care of files and file management, the Windows operating system uses the File Manager. The mission of the File Manager is to organize the files so that users can obtain them quickly and easily.

In the Windows operating system, there are three file systems used on hard drives: there are the NTFS file system, the older FAT and FAT32. Windows also supports file systems for CD-ROM and DVD.

15.1 FAT

FAT is an abbreviation for File Allocation Table, and is the file system used in MS-DOS and early versions of Windows. There have been several versions of FAT, including FAT12, FAT16, FAT32 and exFAT.

FAT12 was a 12-bit address system designed for floppy disks, while FAT16 was developed when PCs with hard disks were taken in use. The first versions of MS-DOS and the very first versions of Windows used FAT12 and FAT16, which are no longer in use.

The early versions of the Windows operating system were Windows 95, Windows 98 and Windows Millennium, all of which used FAT32, and which is no longer in general use.

Although Windows do not use FAT any more, FAT is still in use. FAT is a useful format for solid-state memory cards and it is often used as file system on SD cards.

exFAT, which is also called FAT64, is a newer version of FAT. exFAT is designed for smaller storage devices such as USB pens, so is therefore a version of FAT still in use.

15.2 NT FILE SYSTEM

The NT File System (NTFS) is the file system developed for Windows NT, and is the file system that current versions of Windows use.

NTFS supports long file names, security, fault tolerance, encryption, disk compression and very large files and volumes.

NTFS has several advantages over FAT32 when it comes to safety, reliability, extensibility and efficiency.

Safety

Security is enhanced by the fact that users are given access to just the directories and files they need. In FAT32, all users could access all the files on a hard disk.

Reliability

NTFS keeps track of changes in the file system by keeping a journal. NTFS uses log files to keep track of all disk activity, which allows an NTFS volume to recover quickly after a disk crash.

Extensibility

Using NTFS formatted volumes can expand the storage capacity to existing volumes without having to take backup, to repartition, to reformat or to restore anything.

Efficiency

NTFS volumes will manage partitions larger than 8 GB of memory more efficiently than the old FAT32 file system.

Compression

NTFS supports the compression of files and directories. You can create a file as compressed, and then NTFS automatically compresses the contents of the file.

File names

A file name in NTFS can be up to 255 characters long. File names are in Unicode, which means that you can use file names in character sets other than Latin, e.g. Greek, Chinese, Russian, etc.

15.3 SUPPORT FOR SSD DISKS

Solid State Drive (SSD) is a storage device that uses flash memory instead of mechanical components such as hard drives. This makes SSDs faster and less noisy than the usual hard drives, though the disadvantage of SSDs is that they have been more expensive than hard drives.

Versions of Microsoft Windows prior to 7 did not take any special measures to support solid state drives. Windows 7 has support for SSDs, with the support in Windows 8 and Windows 10 being similar. The operating system detects the presence of an SSD and optimizes its operation accordingly.

15.4 FILE SYSTEMS FOR CD-ROM AND DVDS

Windows supports a file system for CD-ROM called a Compact Disc File System (CDFS), which is the standard for optical disc media. CDFS supports various file systems such as Windows, Mac OS and UNIX systems, so that data can be exchanged between operating systems.

Windows also supports a file system called a Universal Disk Format (UDF). A UDF is a file system standard for data storage on all optical media, but is mostly used for DVDs and newer optical media. For example, Blu-ray and DVD video use different versions of UDF.

15.5 CONTROL QUESTIONS

1. Which versions of Windows used FAT32?
2. What is the use of exFAT?
3. What file system is used by current versions of Windows?
4. What advantages does NTFS have compared to FAT32?
5. What is the difference between SSD disks and traditional hard drives?
6. What versions of Windows have support for SSD disks?
7. Which file systems does Windows support for CD-ROM and DVDs?

16 STORAGE MANAGEMENT

Storage Management defines how operating systems interact with disks and storage media. Windows provides support for many types of storage media, including hard drives, USB drives, tape drives and network storage such as SAN (Storage Area Networks) and iSCSI (Internet Small Computer System Interface).

In the following, we shall look at storage on hard drives, since hard drives have long been the primary storage media for computers.

16.1 ABOUT STORAGE ON HARD DRIVES

A disk consists of sectors, which are a block of a specific size that has an address. A hard drive consists of a series of blocks. The size of the blocks has been from 2 to 8 Kbytes, and the blocks have numbers from zero and upwards.

One task for an operating system is to manage hard drives. When a new file is to be stored, the operating system must allocate space on the hard disk. Some ways to do this are:

1. Interconnecting allocation
2. Linked list
3. File Map
4. Indexed allocation

Interconnecting allocation means that the file is stored in blocks that are in consecutive order, e.g. 1, 2, 3, 4, 5, 6, 7…. The operating system then only needs to keep track of the starting block and the number of blocks that the file occupies.

The problem with the connecting allocation is that:

1. It is not always that contiguous blocks are free on the hard disk when creating a file.
2. When expanding a file, free blocks may not be available so that the expansion will not be continuous.

Another way to keep track of a file is to use a linked list. Each block has a pointer that points to the next block in the file, hence the blocks do not need to n consecutive order. A disadvantage of a linked list is that if a pointer is not positioned correctly to a block, all blocks that follow will be lost.

A file map is an enhancement of the linked list. All the pointers are located in one place, so in this way, we get a map of the file. If one pointer is wrong, it does not influence the others.

Indexed allocation is also an improvement of the linked list, as all the pointers are in one place. The pointers are accessed using indexes instead of using a pointer to the next block in the way of the linked list.

16.2 ORGANIZATION OF HARD DISKS

There are different ways to organize the data storing in a computer system. A PC typically only has one volume on one hard drive. However, it may be beneficial to organize a disk by dividing it into several volumes.

Volumes used by servers will often extend across multiple hard drives, the purpose of this being to increase the reading speed and security.

Partitions

A hard drive consists of one or more partitions, which is a section of a hard drive. Each partition can then function as a separate disk.

A partition is a collection of contiguous sectors on a disk. A partition table or other databases for disk management stores the starting sector and the size of the partitions.

The task of the Partition Manager is to create, delete and manage partitions, thereby ensuring that all partitions have a unique ID.

Volumes

The storage on a computer is divided into volumes designated by a letter such as C, D or E. A simple volume is only one partition, but you can organize volumes so that they consist of multiple partitions on one or more hard drives.

A simple volume uses only one hard drive, which means that if the hard disk crashes, the volume is out of use. Using multiple volumes and multiple disks avoids this, hence you can still access data even if one hard drive is not functioning.

There are two types of volumes:

1. Simple volumes representing sectors on a single partition
2. Multi-partition volumes representing sectors from multiple partitions

Advantages of multi-partition volumes are performance, reliability and the size of volumes. Some types of multi-partition volumes are spanned volume, striped volume, RAID 1 volume and RAID 5 volume.

Spanned Volume

A spanned volume is a volume that consists of multiple partitions.

The advantage of using a spanned volume is that one can expand the volume without having to replace the drive with a new larger drive, whereas the disadvantage of a spanned volume is a greater risk of disk failure when the partitions are located on different drives.

Spanned volumes have no fault tolerance such as RAID, so if a disk fails, the entire volume will be lost.

Striped volume

A striped volume consists of multiple partitions on multiple hard drives. When you write to a striped volume, the data is distributed across all hard drives.

The advantage of a striped volume is that the reading speed increases because the computer can read the data from disks in parallel. The disadvantage of a striped volume is that it can easily cause drive failure.

RAID 1 volume

A RAID 1 volume consists of two hard drives. The same data is on both drives, which means that one disk is a copy of the other.

The advantage of RAID 1 volume is security. If one of the hard drives fails, you still access the data from the other hard drive. The disadvantage of RAID 1 is that it uses twice as much storage space.

RAID 5 and RAID 6 volumes

RAID 5 volumes has been widely used for storing data on servers because it increases both the reading speed and security. To use RAID 5, at least three disks are necessary, as RAID 5 writes the data to multiple disks. Security improves when additional information about the data (parity data) is on multiple disks.

RAID 5 results in an increased reading speed because you can read the data from disks in parallel. However, RAID 5 entails a little slower write speed, as parity data is also written.

RAID 5 can tolerate that one of the disks fails since the other disks will continue to work so that you can retrieve data there. Even so, if a disc is out of use the reading speed will be slower, as you must use the parity data.

RAID 6 is an improvement of RAID 5, and is quite similar to RAID 5. RAID 6 is an alternative to RAID 5 if the storage system contains four disks or more because RAID 6 tolerates that two disks fails.

Standard disks and dynamic disks

Windows distinguishes between two types of disks:

1. Basic Disks
2. Dynamic Disks

A Basic Disk has a fixed size and is only on one physical hard drive. A basic disk can contain several partitions, and there may be several volumes of a standard disk.

Dynamic disks are more flexible than basic disks. A disk must be dynamic to have a shared volume with other disks. It is therefore necessary to use dynamic disks if you use multi-partition volumes such as spanned volume, striped volume or RAID 5 volume. One advantage of dynamic disks is that you can resize a disk without restarting Windows.

16.3 STORING DATA ON SERVERS

Networked computers offer new possibilities for data storage. The organizing of data storage on servers is usually done in another way than for a single PC.

SAN

A SAN (Storage Area Network) is a storage medium that is available to servers via a network so that it looks as if the storage media is located locally on the server.
An advantage of using a SAN is that multiple servers can use a single storage medium, which is called storage sharing. Storage sharing is economical, simplifies management and is more effective if a server crashes.

iSCSI

iSCSI (Internet Small Computer System Interface) is a network standard based on IP (Internet Protocol) to access a storage device via a network. The purpose of iSCSI is to help facilitate data transmission over the network and manage the storage of data over long distances. iSCSI can be used to transmit data over LANs (Local Area Networks), WANs (Wide Area Networks) and the Internet.

iSCSI has become a popular way to store data on a network. This is because it is possible to store data on a remote server in the same manner as a local computer.

Data Storage in a Cloud

Data can be stored in a Cloud. You can store your data on servers that are located at remote server parks connected to the Internet. Instead of having your own storage system in a LAN, you can rent storage space on a server via the Internet.

It is expected that the cloud will greatly influence the IT industry and the use of computers in the future since the cloud can provide a combination of cost effectiveness and flexibility. Instead of having to buy servers themselves, companies can lease capacity when needed.

16.4 CONTROL QUESTIONS

1. What are the advantages and disadvantages of using a connecting allocation when files are stored in a hard drive?
2. Define the following terms associated with hard disks: partition, volume, simple volumes, multiple partition volumes, spanned volume and striped volume.
3. What is the benefit of using RAID 1 volume?
4. What is the reason that RAID 5 volume has been widely used?
5. What is the difference between a basic disk and a dynamic disk?
6. What are the advantages of using SAN as a storage medium?
7. What is iSCSI and why has iSCSI become a popular way to store data?

17 NETWORKING FEATURES IN WINDOWS

The first of Microsoft's operating systems had little network support, although Windows NT was developed to function in a network. In today's Windows, there is broad support for network tasks in the input/output system and the Windows API.

The common task for a network application is to take a request from an application on one computer and send it to another computer. The remote computer then performs the request and sends back the result.

For this to occur, the request must often be transformed several times. For example, a request sent over the Internet will be divided into several packages.

An operating system hence needs services that enable communication between computers. A Windows operating system has networking software just for this purpose.

There are four types of networking software in Windows: network services, network APIs, protocols and drivers for network equipment.

17.1 NETWORK SERVICES

Windows has several network services based on the API components.
Two of these are Remote Access and Active Directory.

Remote Access

Remote Access allows clients to associate a connection to servers so that they can obtain resources via a network connection. This may be resources on a server such as files, printers and network services. Windows allows two types of remote access, which are a dial-up connection and a Virtual Private Network (VPN).

A dial-up connection allows clients to connect to a server via telephone lines or a similar infrastructure. Dial-up is a temporary physical or virtual connection between a client and a server.

Remote Access with VPN establishes a connection to a server over an IP network such as the Internet. You can log on with VPN to a server in local network from anywhere in the world via the Internet.

Active Directory

Active Directory is a tool used to manage a local network using a Windows Server operating system, and is a directory that provides an overview of all users and all devices in the network.

With Active Directory, a network administrator organizes users and computers in groups. In large networks, it is necessary for group users and computers to maintain an overview of the network. Active Directory makes it possible to manage very large networks with up to millions of users.

17.2 SOME NETWORK APIS

Network APIs are network functions in Windows API. Windows has several network APIs that provide support for software, and applications can use these to communicate with programs on other computers.

Some network APIs include Windows Sockets, Remote Procedure Call, Named Pipes and Mailslot's.

Windows Sockets

Windows Sockets API, also called Winsock, is a technical specification that defines how Windows network software should cooperate with network services such as TCP/IP. Windows Sockets provides an interface between a Windows TCP/IP client program and the underlying TCP/IP protocol.

Windows Sockets makes it possible for developers to create advanced network of applications for both the Internet and intranet using Microsoft Windows networking functions, regardless of the network protocol used.

Remote Procedure Call

Remote Procedure Call (RPC) is a type of process communication that allows a computer to execute code on another computer over a network, without the programmer having to code the details of the communication. In other words, the programmer writes about the same code, whether the code is executed on the local computer or on a remotely located computer. RPC therefore makes the communication process as simple as a function call, operating between processes at different computers in a network.

Named Pipes

Named Pipes is a programming API for communication between processes, which takes place between a named pipe server and a named pipe client. A named pipe server is a program that creates a named pipe that clients can use, as data is transferred via a buffer. A process writes data to a buffer so that another process can read the data from the buffer.

In Windows, Named Pipes is a client/server communication that works quite similarly to Sockets.

Mailslot

Mailslot is a broadcast mechanism for one-way communication that allows communication between processes, both locally and over a network. The messages are usually sent via a local network or an Internet network. A process that creates a mailslot is called a mailslot server. Other processes (clients) can send messages to a mailslot server that has a name. Mailslot´s provides a simple way of sending short messages.

17.3 ABOUT NETWORK DRIVERS IN WINDOWS

Network API drivers take API requests and translate them into network protocol requests, so that it is possible to send them via a network. API drivers use transport protocol drivers to do this translation.

In 1989, 3Com and Microsoft developed NDIS (Network Driver Interface Specification), which allows protocol drivers to communicate with network adapter drivers. NDIS is independent of the type of equipment used by a computer. Network adapter drivers that use NDIS are called NDIS drivers or NDIS miniport drivers.

Transport Driver Interface (TDI) is an interface developed by Microsoft to make it easier for drivers to communicate with various network transport protocols. The advantage of using TDI is that services are independent of different protocols for transport in networks.

TDI transports, also called transports and NDIS protocol drivers, are drivers in kernel mode.They receive packets from TDI and send them further.

NDIS miniport drivers are drivers in kernel mode, which provides an interface for TDI transports to network adapters.

About adapters

An adapter is a transition that allows the transfer of data between devices that are not compatible. Adapters are common as transitions between different connections in computer equipment, in which plug-in formats in periods have been non-standardized.

17.4 CONTROL QUESTIONS

1. What kind of network software is there in standard Windows?
2. How does remote access work with VPN?
3. What is the task for Active Directory?
4. What are network APIs?
5. Mention some network APIs in the Windows operating system.
6. What is the role of network API drivers?
7. What is the task of NDIS (Network Driver Interface Specification)?
8. What is the role of TDI (Transport Driver Interface)?

18 SECURITY IN WINDOWS

In a computer system, there is often sensitive data, to which it is not desirable that there should be public access. The operating system must be able to protect files, memory and setup data so that unauthorized persons cannot read or modify the data.

18.1 ABOUT SECURITY IN COMPUTER SYSTEMS

There are four groups of security for computer systems.

1. Data Confidentiality
2. Data Integrity
3. Access to the system
4. Attacks from outside

Data Confidentiality concerns how to prevent unauthorized access to data on computers. This is important for protecting documents to prevent unauthorized persons from gaining access to- or reading them. This applies not only to secret documents, but also personal data that could be sensitive.

Data Integrity is about preventing unauthorized changes to data in files. This applies not only to a change of data, but also to remove or to add false data. An example of this is a student who will attempt to get into the school's computer system to change his/her grade.

Security in terms of access to the system revolves around nobody being allowed to disturb the system or to put it out of operation.

The fourth type of security concerns the preventing of attacks via the Internet, as hackers may attempt to gain control of computers via the Internet. One way to do this is by the use of a virus. By gaining control of a computer, hackers can use the computer for illegal activities or to send e-mails (spam).

18.2 SECURITY IN WINDOWS NT

With Windows NT came the following security in the Windows operating system:

- Secure login
- Access control for files
- Privileged access control
- Address space defense for each process
- Clearance of pages in memory
- Auditing of computer systems

Secure login means that all users use a password to login. Ctrl + Alt + Delete has been used to login, the purpose of the Ctrl + Alt + Delete being that no one should be able to add fake login windows to capture users' passwords.

Access control to files allows a user who owns a file to decide who else can access the file. Privileged access control means that the administrator has the right to determine access to files if needed, i.e. the administrator can change users' rights.

Address space defense for each process means that each process has protected addresses that unauthorized processes cannot access.

The clearance of pages in a memory means that new pages loaded into the memory will not be able to find information left behind by the previous pages. This makes it difficult for spyware to snoop into memory.

Auditing means that the system writes events in a network that may affect safety to a log file. The administrator will then be able to get information about what is happening on the system by reading the log files. The administrator can decide which events to monitor, and such events can be:

- Someone tries to do something on the system that is not allowed.
- Someone attempts to log on several times because the login fails.

18.3 SECURITY MECHANISMS IN WINDOWS

Microsoft has put a lot of resources into making the Windows operating system more secure. The reason is that in recent years there have been more and more attacks against computer systems around the world, and some of these attacks have been successful. Such attacks have managed to put the computer systems of entire countries or large businesses, which can cost society billions of dollars.

The Windows operating system therefore has a highly developed security system, which is based on access control and integrity levels. We will now look at how security system protects Windows processes and data.

Security ID

There is a need to identify devices such as threads, which can perform operations on the system. Instead of using names to identify such devices, the Windows operating system uses a SID (Security ID). A SID is a number, and each SID is unique in the world.

A SID can be assigned to either a user or group of users in a network. When a process starts, the process and the threads run under the user's SID. Other threads will not be able to access the process unless they have a SID with special authorization to do so.

Security Descriptor

Each process has information about its reliability that tells what privileges the user and the process have. Each process has a Security Descriptor attached that a Security Descriptor points to for controlling lists. These checklists contain access information that can deny access for users or groups of users.

Access to Objects

Central to the security of the Windows operating system is the protection of objects. Windows has a comprehensive security model that prevents unauthorized access to objects, which requires that before a thread can have access to an object, it must first specify what actions it will perform on the object.

Objects protected in the Windows operating system include files, hardware devices, mailslot's, pipes, processes, threads, events, Mutex's, semaphores, shared memory, input/output ports, timers, volumes, network shares, services, printers, etc.

18.4 DEFENSE AGAINST SPYWARE

Spyware is a type of software that infiltrates a computer system without the owner being aware of it. Spyware can pick up personal information about users on a PC and send it over the Internet back to the originator of the spyware.

Windows Defender, also known as Microsoft Anti Spyware, is a program from Microsoft that has the function to prevent, remove and isolate spyware in Microsoft Windows.Windows Defender uses two mechanisms to detect spyware:

1. Scanning
2. Real Time protection

Windows Defender scans your computer and control programs against a database of information about spyware.

Real Time Protection is a process that runs in the background and is looking for spyware that tries to install itself or run on your computer.

Windows Defender can also remove ActiveX applications and block programs that start automatically at Windows startup.

Windows Defender is included in Windows Vista, Windows 7, Windows 8 and Windows 10.

In Windows 8, Windows Defender was upgraded to an antivirus program that also looks for common viruses, and not just spyware.

18.5 VIRUS PROTECTION

A virus is a program that makes a copy of itself by attaching itself to another object. Viruses can attach themselves to program files, documents and file system structures, and can run when the program linked to it is executed. Viruses can also be in the memory and spread to the documents that a user opens or saves.

A worm is an independent program that spreads by copying itself from one computer to another. The usual way is via a network or attached to an e-mail. Worms are similar to viruses, though the difference between viruses and worms is not so important in practice.

A computer can do without an antivirus program through careful use. One reason for this is that an antivirus program is only one of many layers of security in a protected network. For example, routers prevent viruses from entering via Web browsers through the Internet. E-mail servers also block viruses.

Earlier versions of the Windows operating system did not have their own antivirus program. However, it has been possible to download it for free over the Internet. Today, Windows has Windows Defender included. Windows Defender is malware protection that helps identify and remove viruses, spyware, and other malicious software. Windows Defender runs in the background and notifies you when you need to take specific action. However, you can use it anytime to scan for malware if your computer isn't working properly or if you clicked a suspicious link online or in an email message.

18.6 THE FIREWALL IN WINDOWS

A firewall is a part of a computer system or network designed to block unauthorized access and to allow authorized access. Both hardware and software can implement firewalls.

The purpose of a firewall is to prevent unauthorized Internet users from accessing local network connected to the Internet, especially intranets. The firewall will investigate all messages entering or leaving the intranet through the firewall, and will also block messages if the safety criteria do not hold.

The firewall in Windows filters both incoming and outgoing packets. All incoming packets to your computer are blocked unless they are a response to a request from your computer, while all outgoing packets from your computer are permitted unless they violate a set rule.

Windows Firewall was first introduced as part of Windows XP Service Pack 2, and later versions of Windows have improved the Firewall.

18.7 WINDOWS UPDATE

An important part of keeping a computer system safe is to obtain the latest upgrades to the operating system. Microsoft is constantly working on new upgrades; this may be updated drivers or improvements of code that have contained faults.

Many upgrades just give improvements in performance and functionality, but some are also security updates to the system.

Windows include Windows Update, which is a program that updates the Windows operating system for computers all over the world once a month. Using automatic updates, a Windows operating system upgrades itself over the Internet without having to use a browser. The upgrade is usually the second Tuesday of the month. However, critical upgrades can take place more often if necessary.

18.8 CONTROL QUESTIONS

1. Mention some reasons why there is a need for security in a computer system.
2. What is the task of a Security ID (SID)?
3. What is the task of the Security Descriptor?
4. What possibilities does a user have to scan the computer for viruses in Windows?
5. What mechanisms does Microsoft Defender use to detect spyware?
6. How does Windows 10 protect against viruses?
7. Explain briefly how the firewall in Windows filters the packets.
8. What is the task for Windows Update and how does Windows Update work?

19 WHEN WINDOWS CRASHES

Many conditions can cause Windows to crash. If there is a system crash, Windows stops and displays the blue screen.

19.1 THE BLUE SCREEN

The blue screen (The Blue Screen of Death) will appear in Windows when it has encountered a critical system failure. The system then goes down to prevent the occurrence of serious faults that could damage the system.

According to Microsoft, the reason Windows shows the blue screen is poorly programmed drivers or hardware that is not working properly. The blue screen can also be a result of memory error, power failure, overheated components or hardware used in an improper way.

Often, the blue screen appears when you install new software or new hardware. For example, if you have installed a new driver and tried to reboot, you can get the blue screen. You will then have the opportunity to undo the installation and get back the old configuration.

A question is why errors cause Windows to crash. Why does the system not ignore the error and just continue? The reason is that the error that causes windows to crash is often part of a larger problem. To let Windows continue will lead to more and more serious errors.

The blue screen has been in all versions of Windows since Windows 3.11 (1993).

19.2 SOME REASONS THAT WINDOWS CAN CRASH

Windows may crash for various reasons. A detailed description of all circumstances that may cause Windows to crash will be very extensive. In the following, we shall therefore only look at a few reasons:

Storage failure

The most common reason that Windows crashes is storage failure (Pool Corruption). Pool corruption can occur when a driver runs into problems because a buffer cannot accommodate all the data it receives. Pool corruption can also occur if a driver writes to a storage place that it had before, but has given up.

Page Fault in memory

An error on a page in memory (Page Fault) can lead to system crash.

Poor or no power supply

A driver or a function associated with the operating system will not work without power. Poor power or a lack of power can thus cause Windows to crash.

Access Violation

An access violation occurs if there is an attempt to write to a page into a memory that is only allowed to read, or if the system tries to read an address that does not exist.

Error in memory

If the Memory Manager detects that a data structure in a memory is corrupt, this can cause a system crash.

Hardware

Errors in hardware can cause system crashes. This includes errors on a disk when the memory manager attempts to read data.

USB pen

If an error occurs while performing an operation on a USB pen, this can lead to a system crash.

Error in the file system

A fatal error in the file system can lead to system crash.

19.3 FILE DUMPING BY SYSTEM CRASH

Windows is set up to try to provide information of the system's status when the system crashes. The system writes memory contents to a file (Memory Dump).

Information about a system crash may occur at three levels.

Complete Memory Dump	A Complete Memory Dump writes all the contents that were in the memory when the crash occurred.
Kernel Memory Dump	A Kernel Memory Dump only provides to read/write pages in kernel mode that was in the memory when the crash occurred.
Small Memory Dump	A Small Memory Dump provides an overview of drivers and running processes and threads that were in the memory when the system crashed.

You can see what recommendations Windows has for dumping memory on your computer. In Windows, you can open the Control Panel and select the following:

System and Security – System – Advanced System Settings

This brings up the System Properties. Choose Settings under Startup and Recovery.

19.4 CONTROL QUESTIONS

1. Under what circumstances will Windows show the blue screen?
2. Mention some conditions that can make Windows crash.
3. Why does Windows ignore an error and just continue, instead of showing the blue screen?
4. How can Windows give information about what caused a system crash?

WORDLIST

The wordlist provides an explanation of some terms of the operating systems used in this presentation.

Asynchronous I/O

Asynchronous input/output lets a program continue to perform other tasks without waiting for an input/output operation to be performed.

Cache

The purpose of caching is to make the input/output more efficiently.
Data is stored in a cache for fast access.

Cache Manager

A Cache Manager is a component in Windows that makes caching services for NTFS and other file systems.

Context switching

The purpose of context switching is to remove a thread from the processor and to load it in a new thread for execution.

Critical Section

A Critical Section is a piece of program code that tries to use a resource that can only be used by one thread at a time.

Deadlock

Deadlock occurs when two threads stop running because both threads are waiting for a resource that the other threads control. There may be more than two threads involved in a deadlock.

Device Driver

A driver is a control program that enables communication between a hardware device and the operating system. For the Windows operating system, a driver is necessary to use and communicate with hardware equipment.

Executive Services

Executive Services in the Windows operating system helps applications in user mode to run on a computer's system. Executive Services include many basic services in the Windows operating system.

Fiber

A fiber is a lightweight thread. The operating system does not schedule fibers like threads, as fibers belong to a thread that schedules them.

Handle

A handle is a type of pointer used in the Windows operating system. User applications do not have direct access to data in objects. Therefore, applications use a handle in order to reach and change data in objects.

Hardware Abstraction Layer

A hardware layer (HAL) is a layer between the physical hardware and software that runs on a computer. A task for HAL is to get the Windows operating system to be able to work with different hardware platforms.

Heap

The heap is a portion of memory allocated to a process. The process can use the heap to store variables and objects associated with the process.

Inter Process Communication

When a thread in one process interacts with a thread in another process it is called an Inter Process Communication.

Interrupts

Events in a computer system can happen at any time, and then the processor must process them. Input/output devices, timers, etc. send interrupts to the processor when they have tasks for the processor.

Job

Windows can group processes that cooperate. Such a group of processes is called a job.

Kernel

The kernel is an essential component that takes care of the operating system's use of the processors.

The kernel controls the use of the processors, in addition to managing process planning, context switching, interrupt signals, exception signals and multiprocessor synchronization.

Kernel Mode

The Kernel Mode is a privileged mode for processes belonging to the operating system. All system resources in the operating system available for processes are running in the Kernel Mode.

Mutual Exclusion

Mutual Exclusion is when one thread has received exclusive control to a resource, so that other threads cannot access the resource.

Multitasking

Multitasking is a technique that the operating system uses to share one processor for multiple threads. The threads take turns in the processor, and a thread is only in the processor a short time before being replaced by another thread.

Object

An operating system object is a data structure that represents a system resource such as a file, a thread, a semaphore, a driver, an event, etc.

Paging

Through paging, the contents of the memory are moved to a hard disk, so that the memory is released to other processes or to the operating system itself. Data is divided into a set of equally sized blocks called pages. The advantage of paging is that you do not have to keep all programs in the memory at the same time, and that you always work with units of the same size.

Process

When a program starts and is loaded into a computer's memory, we have a process.
In Windows, a process is a container in the memory of a program running on the computer.

Processor Scheduling

The task of Processor Scheduling is to handle the sharing of one or more processors among a group of threads.

Quantum

A quantum is a time interval that a thread is allowed to run on the processor before it must exit and give the processor to another thread.

Registry

The registry is a system database that contains information to start Windows and to configure the system. The registry keeps track of system settings and user preferences, as well as information about hardware and applications on your computer.

Scheduler

Scheduling or process planning is taken care of by the Process Scheduler, which is part of the operating system.

Synchronizing

Synchronization is a coordination of threads to avoid collisions. Synchronization ensures that only one thread at a time can access a resource that cannot be shared.

Synchronous I/O

Synchronous input/output will stop a thread to execute when it performs an input/output. The thread will not continue before the input/output operation is finished.

Timer

A timer controls events in an application code. This occurs with function calls at a determinable time interval, for example every second. The code in the function will then be performed.

Thread

A piece of program code to be executed in the processor is called a thread, which is the dynamic part of a process.

User Mode

User applications are running in user mode. For security reasons, applications running in user mode have restricted access to data that belongs to the operating system.

Virtual Memory

Virtual Memory is a technique that provides applications with the impression that your computer has a contiguous memory, even if some of the memory can be physically located on different storage media such as the memory and a hard disk.

Volume

The hard drive(s) on a computer can be divided into volumes that are designated by letters C, D, E… A single volume is only one partition, but a volume can span multiple partitions on one or more hard drives.

SOME REFERENCES

Windows Internals 7. Pavel Yosifovich, Alex Ionescu, Mark E. Russinovich, David A. Solomon

Modern Operating Systems. Andrew S. Tanenbaum

Operating Systems. William Stallings

Windows Inside Out. Ed Bott, Carl Siechert, Craig Stinson

Windows System Programming. Johnson M. Hart