

**COURSE: Design of Operating System (CSE315)**

**Case Study: Windows 10**

**SEMESTER: SUMMER 2021**

**SECTION: 1**

**Faculty: Mohammad Noor Nabi**

**SUBMITED BY:**

**NAME: Israt Jahan**

**MD. Wahid Hassan Fuad**

**MD. Ashikur Rahman**

**ID: 1710718, 1730149, 1831110**

Table of Contents

[**1.** **Introduction:** 2](#_Toc71346662)

[**2.** **Windows 10 Design Goals:** 4](#_Toc71346663)

[**3.** **Windows 10 Structure:** 5](#_Toc71346664)

[**4.** **Components of Windows 10:** 6](#_Toc71346665)

[**5.** **Shells of Windows 10 :** 9](#_Toc71346666)

[**6.1. Process Creation** 11](#_Toc71346667)

[**6.2. Process Termination** 12](#_Toc71346668)

[**6.3. Process Communication** 14](#_Toc71346669)

[**6.** **Process State Diagram** 15](#_Toc71346670)

[**7.** **Process Management System-Calls** 17](#_Toc71346671)

[**8.** **Process versus Thread** 19](#_Toc71346672)

[**10.1** **User level thread Model** 20](#_Toc71346673)

[**10.2** **Kernel-Level Threads** 21](#_Toc71346674)

[**11.** **Process Scheduling Parameters** 22](#_Toc71346675)

[**12.** **Process Scheduling Algorithm** 23](#_Toc71346676)

[**13.** **Synchronization** 25](#_Toc71346677)

[**14.** **Booting Process** 26](#_Toc71346678)

[**15.** **Virtual Memory Management** 27](#_Toc71346679)

[**Extra: Dynamic Device Support** 28](#_Toc71346680)

[**Citations :** 29](#_Toc71346681)

# **Introduction:**

Microsoft Windows, or just Windows, is a collection of proprietary graphical operating system families that are all created and marketed by Microsoft. Each family focuses on a different aspect of the computing industry. Windows NT and Windows IoT are active Microsoft Windows families, with subfamilies such as Windows Server and Windows Embedded Compact (Windows CE). Windows 9x, Windows Mobile, and Windows Phone are all defunct Microsoft Windows family.

**1.1: About Windows Operating System:**

• Windows OS, computer operating system (OS) developed by Microsoft Corporation to run personal computers (PCs). Featuring the first graphical user interface (GUI) for IBM compatible PCs, the Windows OS soon dominated the PC market. Approximately 90 percent of PCs run some version of Windows.

• The first version of Windows, released in 1985, was simply a GUI offered as an extension of Microsoft’s existing disk operating system, or MS-DOS. Based in part on licensed concepts that Apple Inc. had used for its Macintosh System Software, Windows for the first time allowed DOS users to visually navigate a virtual desktop, opening graphical “windows” displaying the contents of electronic folders and files with the click of a mouse button, rather than typing commands and directory paths at a text prompt.

• Subsequent versions introduced greater functionality, including native Windows File Manager, Program Manager, and Print Manager programs, and a more dynamic interface. Microsoft also developed specialized Windows packages, including the networkable Windows for Workgroups and the high-powered Windows NT, aimed at businesses. The 1995 consumer release Windows 95 fully integrated Windows and DOS and offered built-in Internet support, including the World Wide Web browser Internet Explorer.

• With the 2001 release of Windows XP, Microsoft united its various Windows packages under a single banner, offering multiple editions for consumers, businesses, multimedia developers, and others. Windows XP abandoned the long-used Windows 95 kernel (core software code) for a more powerful code base and offered a more practical interface and improved application and memory management. The highly successful XP standard was succeeded in late 2006 by Windows Vista, which experienced a troubled rollout and met with considerable marketplace resistance, quickly acquiring a reputation for being a large, slow, and resource-consuming system. Responding to Vista’s disappointing adoption rate, Microsoft developed Windows 7, an

OS whose interface was like that of Vista but was met with enthusiasm for its noticeable speed improvement and its modest system requirements.

**1.2: History of Windows Operating System (Timeline):**

Microsoft’s Windows operating system was first introduced in 1985.

A picture containing timeline

Description automatically generated

Figure: Timeline of Windows Operating System

**1.3: Windows 10:**

Windows 10 is a series of Microsoft operating systems that were introduced as part of the Windows NT operating system family. It is the successor to Windows 8.1, released nearly two years earlier, and was released to manufacturing on July 15, 2015, and broadly released for the public on July 29, 2015.

Windows 10 represents another step in Microsoft’s U-turn, bringing back the Start menu and more balance to traditional desktop computer users.

# **Windows 10 Design Goals:**

**1. Designed for Tablets**

Tablet support is arguably the most essential new feature in Windows 10. It outperforms the similar feature in Windows 8.1.

**2. A streamlined update process**

If there is anything that annoys Windows users, it is the requirement to update the software often and then restart to apply those updates. Microsoft changed that by consolidating all restarts related to updates into a single, monthly reboot. The result, Microsoft says, is less "disruptiveness."

**3. Major security enhancements**

Unfortunately, Windows has not always been the most secure operating system. But with Windows 10, there is a strong possibility it could set a new security benchmark for the Windows platform. Much of that is due to how Microsoft reworked security features at the operating system's lowest level. The company has also been able to make the software more adept at finding malware and preventing it from running on the PC.

**4. The app store**

The features of the app store are more enhanced and well balanced with the environment in windows 10.

**5. Improved energy efficiency**

Microsoft has said that it understands consumers and enterprise users are going mobile. Windows 10 has come with improved energy efficiency features that should make notebook and tablet batteries last longer. Windows 8.1 can be a power hog because of its resource intensiveness. Microsoft is fixing that with Windows 10.

**6. Improved memory usage**

It seems that over the years, more and faster RAM has become a big concern for PC buyers. But Microsoft tried to limit that with Windows 10 by making the operating system more capable of handling multiple applications and processes without using up too much RAM.

**7. Improved cloud-based service**

Although Microsoft has several cloud-based applications, it has historically been loath to embrace that in Windows. In Windows 10, however, the company is doing just that by allowing the operating system to tap into Windows Live SkyDrive, a cloud data storage service. Windows

is far from being cloud-based, but at least it is starting to rely somewhat on the cloud to extend its functionality.

# **Windows 10 Structure:**

**3.1: Windows Core:**

For the first time, the Windows core contains a standard set of APIs that provide complete binary compatibility with a wide range of systems. For almost a decade, Microsoft has been launching whole operating systems, and there were numerous similarities between them, such as Windows Phone, Windows CE, and Windows RT.Most of them were same, but the underlying layer was different. The Windows Core now is that common refactored core that is common for all the Windows Releases be it for Mobile or Desktops.  
  
Windows Core is very small and reduced subset of all the versions of windows. This is done to keep in mind that Windows can now have one single codebase rather than a shared codebase and the usage on an entire range of various devices that range from IoT to Desktops.

**3.2 The converged Kernel Approach:**

In creating Windows 10, the converged Kernel approach made a huge influence. The new look and feel first appeared on the Xbox 360 and Windows 7.5, which was later changed in Xbox One (with the addition of a dashboard and other features), and with Windows Phone 8, the look and feel was fully integrated.But there was a big difference underneath that lead the users and developers to Windows 8.1 for Mobile and Desktops. This created the Converged App Model and resulted in the possibility of the development of “Universal Apps”. But now with Windows 10, we not only have the Kernel in common, but also all the fundamental pieces of the OS. Also, the applications that were Universal Apps had two separate binaries whereas now the Windows Core has one single binary across all platforms.

Diagram

Description automatically generated

Figure: Converged Kernel Approach

**3.3: Universal App Platform:**

The Universal App Platform (UAP) is a collection of contracts and versions. This is the new platform that developers will use to create their applications. The UAP is based on the Windows Core, which is a set of Low-Level APIs organized into Contracts.

# **Components of Windows 10:**

[](http://www.whenparentstext.com/wp-content/uploads/2019/11/MicrosoftVista-4x3.jpg)

Windows 10 consists of the following components:

• **Desktop:**

The desktop is the first screen we see after Windows starts. Here, we can see folders such as "My Computer" and "Documents". It is the main work area for many computer operations. It saves your special files on the desktop so that you can easily access them, and gives you access to other important functions, such as the search bar, task bar, and file browser.

• **Taskbar:**

The taskbar is a simple row at the bottom of the screen that lists all of the currently open files and applications. It guides you through the process of deciding what you want to keep open and what you want to close.

• **Start Menu:**

By clicking the start menu, in the bottom left corner of the screen, a vertical window consisting of the recently opened applications and saved locations will pop-up. Although the Start Menu was a major component of Windows before Windows 8, It was removed from Windows 8 and then brought back in Windows 10.

• **Maximize/Minimize/Close Buttons:**

These buttons are located at the top right corner of our opened documents, and the area used to close, minimize, or maximize the document window. They help us jump from one task to another fast and let us decide either we want to close an application or resize its area on the screen or just hide it for a few moments.

• **My Computer:**

When we double click on My Computer, we find ourselves looking at a window where we can navigate between Computer Drives and Control Panel tools. It also gives us access to different drives on our computer and the data which lies in those drives.

• **My Computer Right Click Menu:**

When we right-click on My Computer or any other file or folder, we get a menu where we can look into different options related to that specific file, for example, Properties, etc.

• **Recycle Bin:**

When we delete a file or folder, it goes into the Recycle Bin from where It can either be restored or permanently deleted from the Computer. Once, a file or folder is deleted from the Recycle Bin, it is very difficult to recover it again. Therefore, the utility of the recycle bin is very essential to use properly if users deal with important documents and files on a day to day basis.

• **Shortcut:**

A shortcut creates a button or icon which typically is located on the Desktop. By clicking on this Shortcut, we can quickly open the document or application of which it is a shortcut. It helps us save the tedious task of going to the main directory again and again and saves our time.

• **Mouse Functions:**

The mouse is an input device which is essential in the working of a computer. It performs several important functions on Windows like Scrolling, Right and Left Clicks, etc. It performs another very important function of modern windows which is pointing towards different things and giving special instructions whenever needed.

• **Highlight:**

When we have opened a document, we can easily highlight the required portion of our document by using Mouse. It is essential for documents and helps keep track of useful information.

• **Copy/Cut/Paste:**

These options are one of the most essential components of Windows. The copy is used to copy a portion of a document from one document to another or a file or folder from one location to another. The paste is used to paste the copied item on the desired location. While Cut is used to move an item to our desired location in the Computer.

• **Toolbar:**

The toolbar is a simple row where we can see different options to customize the look of our opened window. It has two types, Formatting Toolbar, and Standard Toolbar. The standard toolbar consists of options like new documents, save a document, etc. While Formatting Toolbar consists of options like font size, font type, etc.

• **Drag/Drop:**

Dragging an object means to move an object (file or folder) from one location to another and when we reach our desired location, then we can drop the object to that location. It is one of the most used features of windows as users have to move files from one location to another.

• **File Extensions:**

File extensions are used to define the type of the file. For example, an image file will have an extension of .jpg, .jpeg and a Word document will have an extension .docx, .xls, .txt etc. Users could have different types of extensions and these extensions help you decide the type of software that will be used to access these files.

• **Multitasking:**

The term Multitasking means to run more than one file or application on Windows at the same time. It is a very important component of Windows which saves our time as well as allow us to perform more tasks at the same time.

• **Virtual Keyboard:**

A virtual keyboard is a software through which we can see a keyboard on our screen and use it by our Mouse. It is mostly used in the cases when keyboard is not working properly, or one is using windows on a touch device.

• **Disk Drives:**

Disk Drives are drives used to store applications and files. Hard Drives and Floppy Drives are used for this purpose. They are very important for the user’s instructions as well as your hardware to work properly.

• **Defragmenting Hard Drives:**

Defragmenting a Drive means to erase all the data from that drive. It is also another important component of windows as users need to clean up their hard drives from time to time and it also comes as a built-in utility.

# **Shells of Windows 10 :**

The Windows shell is the graphical user interface for the Microsoft Windows operating system. Its readily identifiable elements consist of the desktop, the taskbar, the Start menu, the task switcher, and the AutoPlay feature. On some versions of Windows, it also includes Flip 3D and the charms. In Windows 10, the Windows Shell Experience Host interface drives visuals like the Start Menu, Action Center, Taskbar, and Task View/Timeline. However, the Windows shell also implements a shell namespace that enables computer programs running on Windows to access the computer's resources via the hierarchy of shell objects.

**Desktop**

Windows Desktop is a full-screen window rendered behind all other windows. It hosts the user's wallpaper and an array of computer icons representing:

1. Files and folders: Users and software may store computer files and folders on Windows desktop. Naturally, on a newly installed version of Windows, such items do not exist. Software installers commonly place files known as shortcuts on the desktop, allowing users to launch installed software. Users may store personal documents on the desktop.
2. Special folders: Apart from ordinary files and folders, special folders (also known as "shell folders") may appear on the desktop. Unlike ordinary folders, special folders do not point to an absolute location on a hard disk drive. Rather, they may open a folder whose location differs from computer to computer (e.g. Documents), a virtual folder whose contents is an aggregate of several folders on disk (e.g. Recycle Bin or Libraries) or a folder window whose content is not files, but rather user interface elements rendered as icons for convenience (e.g. Network). They may even open windows that do not resemble a folder at all (e.g. Control Panel).

**Taskbar**

Windows taskbar is a toolbar-like element that by default, appears as a horizontal bar at the bottom of the desktop. It may be relocated to the top, left or right edges of the screen. Starting with Windows 98, its size can be changed. The taskbar can be configured to stay on top of all applications or to collapse and hide when it is not used. Depending on the version of operating system installed, the following elements may appear on the taskbar respectively from left to right:

1. Start button: Provides access to the Start menu. Removed in Windows 8 (but can be added using third-party software), in favor of the Start charm (see below), only to be reinstated in Windows 8.1. Pictured as a Windows logo.
2. Quick Links menu: Invoked by right-clicking on the Start button. Grants access to several frequently used features of Windows, such as accessing the desktop, Settings, Windows Command Processor, Windows Power Shell, and File Explorer.
3. List of open windows: Along the length of the taskbar, open windows are represented by their corresponding program icons. And once pinned, they will remain even after their respective windows are closed..
4. Shortcuts: An update to Windows 95 and Windows NT 4 added a Quick Launch Bar that can hold file, program, and action shortcuts, including by default the "show desktop" command.
5. Deskbands: Toolbars provided by Windows or other programs for easier access to that program's functions.
6. Notification area: Allows programs to display icons representing their status as well as pop-up notifications associated with those icons. By default, Windows volume control, network status, Action Center, date, and time are displayed in this area.
7. "Show desktop" button: Allows users to access their desktops. It is moved from the left of the Taskbar as a Quick Launch shortcut to the rightmost side as its own dedicated hover button in Windows 10
8. Task View: A function in Windows 10 allowing the user to view and manage open windows and virtual desktops. The 1803 version includes the Timeline, adding the ability to view and open previously used apps over a certain period. Task View can be accessed by pressing the Task View button on the taskbar, or by pressing Windows Key+Tab on the keyboard.
9. Cortana and Search: User can utilize Microsoft's Cortana Virtual Assistant, which enables internet searches, searches for apps and features on the PC, and searches for files and documents. Cortana can be accessed by clicking the search bar, pressing the microphone button, saying "Hey Cortana", or by pressing Windows Key+C on the keyboard. Searches can be initiated by also pressing the search bar, or by pressing Windows Key+Q on the keyboard.
10. Action Center: Introduced in Windows 7, the Action Center gave notifications and tips on boosting computer performance and security. In Windows 10, the Action Center serves as a place for all notifications to reside, as well as the location of frequently used settings, such as screen brightness, wireless connectivity, VPNs, Bluetooth, projector connections, and wireless display connections. Replacing the Charms from Windows 8, the Windows 10 Action Center can be accessed by pressing the speech bubble icon on the taskbar, pressing Windows Key+A on the keyboard, or, if using a touchscreen, swiping from the right.

**Task switching**

Windows 10 has a unified task switcher called Task View, which manages not only application windows but virtual desktops as well.

**Start menu**

Starting with Windows 95, all versions of Windows feature a form of Start menu, usually by this very same name.

# **6.1. Process Creation**

The fundamental Windows process management function is CreateProcess, which creates a process with a single thread. It is necessary to specify the name of an executable program file as part of the CreateProcess call.

It is common to speak of *parent* and *child* processes, but these relationships are not actually maintained by Windows. It is simply convenient to refer to the process that creates a child process as the parent.

CreateProcess has ten parameters to support its flexibility and power. Initially, it is simple to use default values. Just as with CreateFile, it is appropriate to explain all the CreateProcess parameters. Related functions then become easier to understand.

Due to Windows’s layered architecture and the presence of environment subsystems, process creation is quite complex. An example of process creation in the Win32 environment under Windows 10 is as follows.

* AWin32 application calls CreateProcess().
* Anumber of parameter conversions and behavioral conversions are done from the Win32 world to the NT world.
* CreateProcess () then calls the NtCreateUserProcess() API in the process manager of the NT executive to actually create the process and its initial thread.
* The process manager calls the object manager to create a process object and returns the object handle toWin32. It then calls the memory manager to initialize the address space of the new process, its handle table, and other key data structures, such as the process environment block (PEBL) (which contains internal process management data).
* The process manager calls the object manager again to create a thread object and returns the handle to Win32. It then calls the memory manager to create the thread environment block (TEB) and the dispatcher to initialize the scheduling attributes of the thread, setting its state to initializing.
* The process manager creates the initial thread startup context (which will eventually point to the main() routine of the application), asks the scheduler to mark the thread as ready, and then immediately suspends it, putting it into a waiting state.
* A message is sent to the Win32 subsystem to notify it that the process is being created. The subsystem performs additionalWin32-specific work to initialize the process, such as computing its shutdown level and drawing the animated hourglass or “donut” mouse cursor.
* Back in CreateProcess(), inside the parent process, the ResumeThread() API is called to wake up the process’s initial thread. Control returns to the parent.
* Now, inside the initial thread of the new process, the user-mode link loader takes control (inside ntdll.dll, which is automatically mapped into all processes). It loads all the library dependencies (DLLs) of the application, creates its initial heap, sets up exception handling and application compatibility options, and eventually calls the main() function of the application.

# **6.2. Process Termination**

When you start an app, the operating system creates a process for an executable file of the app. It contains the program code and its current activity. Windows assigns a special number known as Process Identifier (PID) which is unique for every process. There are a number of reasons you might want to kill a process, and different methods can be used to terminate it. Here is how it can be done.

**Task Manager**

1. Open Task Manager.
2. Click on "More details" in the bottom right corner to enter Full view mode.
3. Select the desired app in the app list.
4. Click on the End task button or hit the Del key on the keyboard.

This is Task Manager's most well known method. The same can be done from the Details tab. It is a special tab which lists process names instead of app names. Select a process in the list and either click on the End process button or hit the Del key. Using the End Task button means Windows first tries to see for a certain timeout if the process has really stopped responding, and attempts to collect a crash or memory dump of the process. It then terminates the app.

**Task kill**

Another classic method to close a process is the console tool taskill. TaskKill is a Windows console tool which can shut down processes from the command line, batch files, scripts and shortcuts. The program uses the same techniques as Task Manager, so won’t work in every situation, but it’s still handy for automating shutdowns. Some processes are running as Administrator (elevated). In order to kill them, it need to open an elevated command prompt instance.

1. Open the command prompt as the current user or as Administrator.
2. Type tasklist to see the list of running processes and their PIDs. Since the list might be very long, a pipe character with the more command can be used.
3. To kill a process by its PID, type the command: taskkill /F /PID pid\_number
4. To kill a process by its name, type the command: taskkill /IM "process name" /F

Taskkill supports many useful options which can be used to terminate apps.

**Power Shell**

To kill a process which runs elevated, need to open PowerShell as Administrator.

1. Open PowerShell. If required, run it as Administrator.
2. Type the command Get-Process to see the list of running processes.
3. To kill a process by its name, execute the following cmdlet: Stop-Process -Name "ProcessName" -Force
4. To kill a process by its PID, run the command: Stop-Process -ID PID –Force

# **6.3. Process Communication**

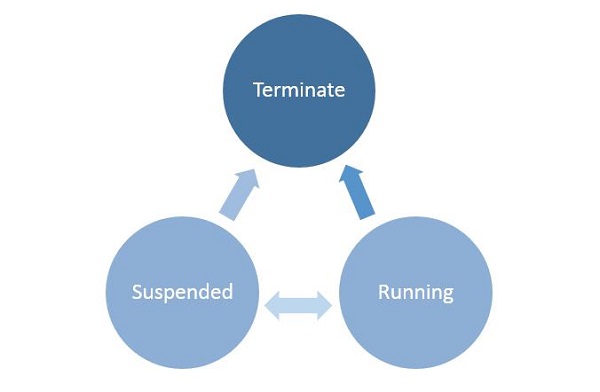
The Windows operating system provides mechanisms for facilitating communications and data sharing between applications. Collectively, the activities enabled by these mechanisms are called interprocess communications (IPC). Some forms of IPC facilitate the division of labor among several specialized processes. Other forms of IPC facilitate the division of labor among computers on a network. After it is decided whether application would benefit from IPC, it should be decided which of the available IPC methods to use. It is likely that an application will use several IPC mechanisms. The answers to these questions determine whether an application can benefit by using one or more IPC mechanisms.

The following IPC mechanisms are supported by Windows:

* **Clipboard:** A loosely coupled data sharing method. When user uses copy or cut command in any application/windows, the copied data is saved in clipboard by windows (temporary storage). The other application can access the data from the clipboard.
* **COM:** Component Object Model offers a platform to interact in Server and Client pattern between processes. COM server can be a local server or In-Process server. There can be also multiple COM clients which interact with the COM server and exchange data.
* **Data Copy:** Windows provides a message i.e. WM\_COPYDATA which enables a process to share data with another process. It can be used with SendMessage API of win32 and COPYDATASTRUCT is used as a parameter. This message is used in case of local computer only.
* **DDE:** Dynamic Data Exchange is a protocol that contains a set of guidelines and rules to send data across processes. A process can use SendMessage API with WM\_DDE\_INITIATE or WM\_DDE\_ACK message sent in response to WM\_DDE\_INITIATE message. It uses shared memory to exchange data.
* **File Mapping:** File Mapping, a fast communication mechanism between processes and gives an efficient way to use the file content in the virtual memory or by accessing the memory sharing. In this IPC the data of the file is treated as a part of the address space of the process so that process can easily access the address of the content. Any other process with access to the shared memory should implement synchronization to mitigate the risks of data getting corrupted. File Mapping is done on the same system/machine and is not available for network processes.
* **Mailslots:** Mailslot provides only one way communication until users create multiple mailslots. One process can create a mailslot as server and the other process create the client mailslot. Mailslot client sends the message to the mailslot server by writing a message and the messages are appended to the mailslot server until the server has read them. A process can have both server and client mailslot and this helps in multi directional communication. Mailslot provides the facility to broadcast the message over the network.
* **Pipes:** Pipes can be used as both single and bi-directional data sharing mechanism. Windows supports two types of pipes i.e. pipe and anonymous pipe. Anonymous pipes can be used in the same network or between the related processes only, while named pipe can be used over a network within different processes. Pipe can be considered as a FIFO queue where one end acts as a server and other as the client.
* **RPC:** Remote Procedure Call provides a way to communicate over the network so that a process can invoke a function in the other process. RPC maintains a tightly coupled relationship between the client and server with high performance.
* **Windows Sockets:** Socket is an efficient way to send and receive data over the network and on the local computer. It uses multiple protocols like TCP/IP and UDP. It is used with the combination of machine IP and Port address where the data can be transported.

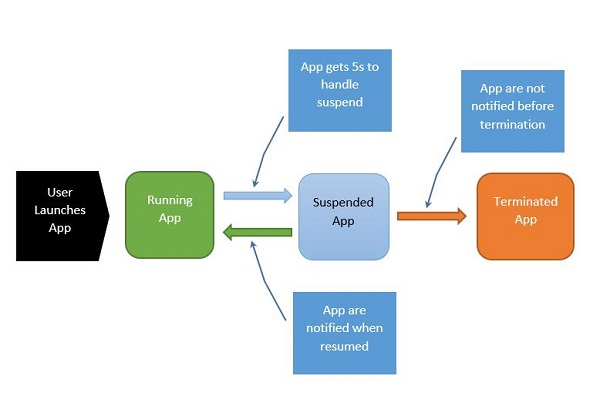
# **Process State Diagram**

Windows applications can exist in three states at the basic level as shown below.

* Running
* Suspended
* Terminate

**Figure : Basic State Diagram**

* When a user launches/activates any application, then it goes in the running state.
* Applications can be suspended if a user does not use it and it is no longer in the foreground.
* From the Suspended state, applications can either resume that application or terminate the OS in order to reclaim system resources.

It is important to understand the process state transitions in a running application. When the user first launches the application, the splash screen is shown and then the application starts running.

**Figure : Process State Diagram**

The process can be explained as follows −

* When the application is suspending, the app gets five seconds to handle that suspended event.
* When the application is suspended, absolutely no code runs and no resources are allocated.
* When it resumes, the app is notified that it has resumed. If it is coming from a suspended state, need to take no action.
* Under memory pressure, it is possible for the application to be terminated.
* Remember that user will not be notified at that point, and so any saving user do, user have to do when user enter into the suspended application state.

When the application transits back and forth between Running and Suspended states, fire suspending and resuming events respectively.

# **Process Management System-Calls**

System calls in Windows 10 are used for file system control, process control, interprocess communication, main memory management, I/O device handling, security etc. The programs interact with the Windows operating system using the system calls. Since system calls are the only way to access the kernel, all the programs requiring resources must use system calls.

Details about some of the important system calls in Windows are given as follows –

|  |  |
| --- | --- |
| **System Call** | **Description** |
| CreateProcess() | A new process is created using this command |
| ExitProcess() | This system call is used to exit a process. |
| CreateFile() | A file is created or opened using this system call. |
| ReadFile() | Data is read from the file using this system call. |
| WriteFile() | Data is written into the file using this system call. |
| CloseHandle() | This system call closes the file currently in use. |
| SetTimer() | This system call sets the alarm or the timer of a process |
| CreatePipe() | A pipe is created using this system call |
| SetFileSecurity() | This system call sets the security for a particular process |
| SetConsoleMode() | This sets the input mode or output mode of the console’s  input buffer or output screen buffer respectively. |
| ReadConsole() | This reads the characters from the console input buffer. |
| WriteConsole() | This writes the characters into the console output buffer. |

**Table: Windows 10 System Calls**

# **Process versus Thread**

Each process provides the resources needed to execute a program. A process has a virtual address space, executable code, open handles to system objects, a security context, a unique process identifier, environment variables, a priority class, minimum and maximum working set sizes, and at least one thread of execution. Each process is started with a single thread, often called the primary thread, but can create additional threads from any of its threads.

A thread is the entity within a process that can be scheduled for execution. All threads of a process share its virtual address space and system resources. In addition, each thread maintains exception handlers, a scheduling priority, thread local storage, a unique thread identifier, and a set of structures the system will use to save the thread context until it is scheduled. The thread context includes the thread's set of machine registers, the kernel stack, a thread environment block, and a user stack in the address space of the thread's process. Threads can also have their own security context, which can be used for impersonating clients.

Microsoft Windows supports preemptive multitasking, which creates the effect of simultaneous execution of multiple threads from multiple processes. On a multiprocessor computer, the system can simultaneously execute as many threads as there are processors on the computer.

A job object allows groups of processes to be managed as a unit. Job objects are namable, securable, sharable objects that control attributes of the processes associated with them. Operations performed on the job object affect all processes associated with the job object.

An application can use the thread pool to reduce the number of application threads and provide management of the worker threads. Applications can queue work items, associate work with waitable handles, automatically queue based on a timer, and bind with I/O.

User-mode scheduling (UMS) is a lightweight mechanism that applications can use to schedule their own threads. An application can switch between UMS threads in user mode without involving the system scheduler and regain control of the processor if a UMS thread blocks in the kernel. Each UMS thread has its own thread context instead of sharing the thread context of a single thread. The ability to switch between threads in user mode makes UMS more efficient than thread pools for short-duration work items that require few system calls.

A fiber is a unit of execution that must be manually scheduled by the application. Fibers run in the context of the threads that schedule them. Each thread can schedule multiple fibers. In general, fibers do not provide advantages over a well-designed multithreaded application. However, using fibers can make it easier to port applications that were designed to schedule their own threads.

# **User level thread Model**

The user-level threads are implemented by users and the kernel is not aware of the existence of these threads. It handles them as if they were single-threaded processes. User-level threads are small and much faster than kernel level threads. They are represented by a program counter(PC), stack, registers and a small process control block. Also, there is no kernel involvement in synchronization for user-level threads.

**Advantages of User-Level Threads**

Some of the advantages of user-level threads are as follows −

* User-level threads are easier and faster to create than kernel-level threads. They can also be more easily managed.
* User-level threads can be run on any operating system.
* There are no kernel mode privileges required for thread switching in user-level threads.

**Disadvantages of User-Level Threads**

Some of the disadvantages of user-level threads are as follows −

* Multithreaded applications in user-level threads cannot use multiprocessing to their advantage.
* The entire process is blocked if one user-level thread performs blocking operation.

# **Kernel-Level Threads**

Kernel-level threads are handled by the operating system directly and the thread management is done by the kernel. The context information for the process as well as the process threads is all managed by the kernel. Because of this, kernel-level threads are slower than user-level threads.

**Advantages of Kernel-Level Threads**

Some of the advantages of kernel-level threads are as follows −

* Multiple threads of the same process can be scheduled on different processors in kernel-level threads.
* The kernel routines can also be multithreaded.
* If a kernel-level thread is blocked, another thread of the same process can be scheduled by the kernel.

**Disadvantages of Kernel-Level Threads**

Some of the disadvantages of kernel-level threads are as follows −

* A mode switch to kernel mode is required to transfer control from one thread to another in a process.
* Kernel-level threads are slower to create as well as manage as compared to user-level threads.

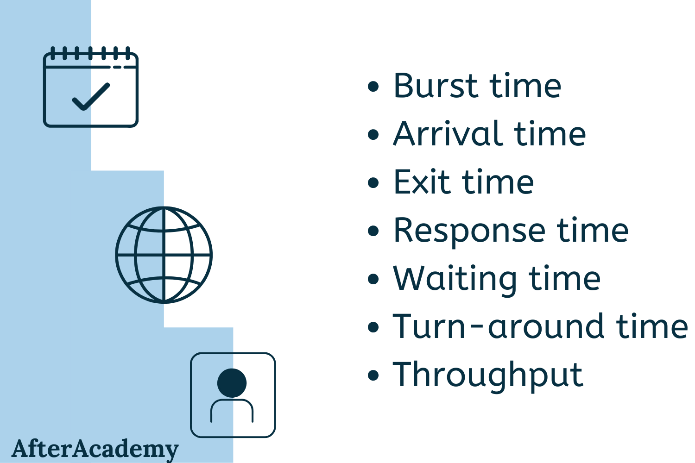
# **Process Scheduling Parameters**

Parameters influence the scheduling of a process. Windows 10 scheduling parameters are dependent on the maintenance plan type. Focusing on Time Based Maintenance Plans and Performance Based Maintenance Plans. The parameters are set to optimize the criteria’s listed below:

* Maximum CPU utilization
* Maximum throughput
* Minimum turnaround time
* Minimum waiting time
* Minimum response time

There are several different criteria to consider when trying to select the "best" scheduling algorithm for a particular situation and environment, including:

* **CPU utilization** **-** Ideally the CPU would be busy 100% of the time, so as to waste 0 CPU cycles. On a real system CPU usage should range from 40% (lightly loaded) to 90% (heavily loaded).
* **Throughput -** Number of processes completed per unit time. May range from 10 / second to 1 / hour depending on the specific processes.
* **Turnaround time -** Time required for a particular process to complete, from submission time to completion.
* **Waiting time -** How much time processes spend in the ready queue waiting their turn to get on the CPU.
* **Load average -** The average number of processes sitting in the ready queue waiting their turn to get into the CPU. Reported in 1-minute, 5-minute, and 15-minute averages by "uptime" and "who".
* **Response time -** The time taken in an interactive program from the issuance of a command to the commence of a response to that command.
* In general one wants to optimize the average value of a criteria (Maximize CPU utilization and throughput, and minimize all the others.) However sometimes one wants to do something different, such as to minimize the maximum response time.
* Sometimes it is most desirable to minimize the variance of a criteria than the actual value. I.e. users are more accepting of a consistent predictable system than an inconsistent one, even if it is a little bit slower.

The processes will have priority values set to it, depending on the parameters. The parameters provide the information on how the scheduler will handle the process. Windows 10 uses this information to optimize the performance, according to its predefined maintenance plan. For example, interactive processes have a higher priority value than batch processes. This is to ensure response time is minimized and acceptable for users, according to Windows 10 standards, for interactive processes.

Process scheduling parameters in Windows 10 are also used to modify existing processes. This is useful when processes have to be transferred to different queues in the system. Multiple queues are available, to be suitable for different processes. Methods are used to determine when a process have to be modified and transferred to a different queue, with a different scheduling algorithm. As processes spend more time without completely being executed, the parameters are used to assign a higher priority to the process, which is also known as aging.

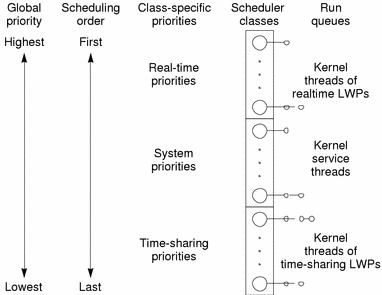
Parameters can be used to set time for a process to be ready. For example, Windows 10 uses scheduling to check for updates periodically. These processes can also be rescheduled to different time intervals, using methods.

Schedules can allow for breaks, for processes to not be available for execution in the set amount of time.

# **Process Scheduling Algorithm**

Windows 10 uses Round Robin and different levels of priority for process scheduling. Multitasking is controlled by the system using the priority values, as processes compete with each other for processor time.

The priority level ranges from 0 (lowest priority) to 31 (highest priority). The highest priority processes are separated from the lowest priority processes, using queues. Higher priority queues have more time slices than other queues. This allows for more processing time of processes that needs to be executed faster.

**Priority Class**

Processes belong to one the classes given below:

* IDLE\_PRIORITY\_CLASS
* BELOW\_NORMAL\_PRIORITY\_CLASS
* NORMAL\_PRIORITY\_CLASS
* ABOVE\_NORMAL\_PRIORITY\_CLASS
* HIGH\_PRIORITY\_CLASS
* REALTIME\_PRIORITY\_CLASS

Processes that are continuously running in the background use IDLE\_PRIORITY\_CLASS. Threads of these processes do not interfere with higher priority threads.

NORMAL\_PRIORITY\_CLASS is the default priority level for all processes at first.

HIGH\_PRIORITY\_CLASS are costly operations. Too many high priority level processes could result in starvation for other lower priority processes. It should be used to execute for a brief time.

REALTIME\_PRIORITY\_CLASS are used for processes that need to be responsive. For tasks such as keyboard and mouse input, disk flushing, refreshing.

**Context Switch**

In round robin, when a process has been executed and the processor is free, Windows looks for the ready queue with the highest priority to be executed next. This is known as Context Switch. The steps of context switch are given below:

1. Save the context of the process that has just been executed.
2. Place the process executed, at the end of the queue.
3. Find the highest priority queue that contains the ready queue.
4. Remove the process at the head of the queue, load its context, and execute.

Lower priority queues can cease to execute, if a higher priority queue becomes available to run. These executions are halted with the Suspend Thread or SwitchToThread functions. The scheduler does not allocate any processor time to the suspended or blocked queues, regardless of their priority. When the processes are ready to be executed, it is transferred to the ready queue.

Common reasons for context switching are given below:

* The time slice has expired.
* A queue with a higher priority has become ready to be executed.
* A running thread needs to wait.

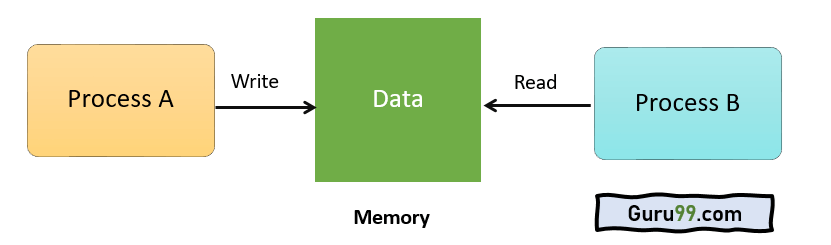
**Priority Boosts**

The scheduler in Windows 10 uses priority values to determine the next executable. Each thread has a dynamic priority and a base priority. Initially all the threads have the same priority value. The system can boost or lower the dynamic priority value, depending on the circumstances. This allows the system to be responsive and ensures process starvation from processor time.

Reasons for dynamic priority boosts are given below:

* When a process is brought to the foreground.
* When a window receives input, such as keyboard inputs, mouse input, timer messages.
* When a blocked thread’s wait condition is satisfied.

# **Synchronization**

****

Windows uses multiple threads to access a resource. This is achieved with the help of a synchronization object and the wait functions. Wait functions can be specified to include the handle of the synchronization object. This means that processes can share the same handle, making it possible for interprocess synchronization.

There are four exclusive types of synchronization objects:

1. Event: Notifies when an event has occurred.
2. Mutex: Can only be owned by one thread at a time, allowing mutually exclusive access to a shared resource.
3. Semaphore: Maintains a count, limiting the number of threads able to access a resource simultaneously.
4. Waitable timer: Notifies waiting threads when a specified time has arrived.

# **Booting Process**

Windows 10 have 4 main stages of the booting process.

1. Preboot
2. Windows Boot Manager
3. Windows Operating System Loader
4. Windows NT OS Kernel

During every process, a program is loaded. The files and file path changes, depending on whether the system uses Legacy BIOS or UEFI. UEFI or Unified Extensible Firmware Interface, is a BIOS replacement that helps set up the hardware and load the operating system.

**Preboot**

When the computer is first turned on, it performs a Power On Self Test (POST). The POST checks for a valid disk system. It also checks for a valid MBR (Master Boot Record). If the process does detect any errors, it moves forward to the next process, windows boot manager.

**Windows Boot Manager**

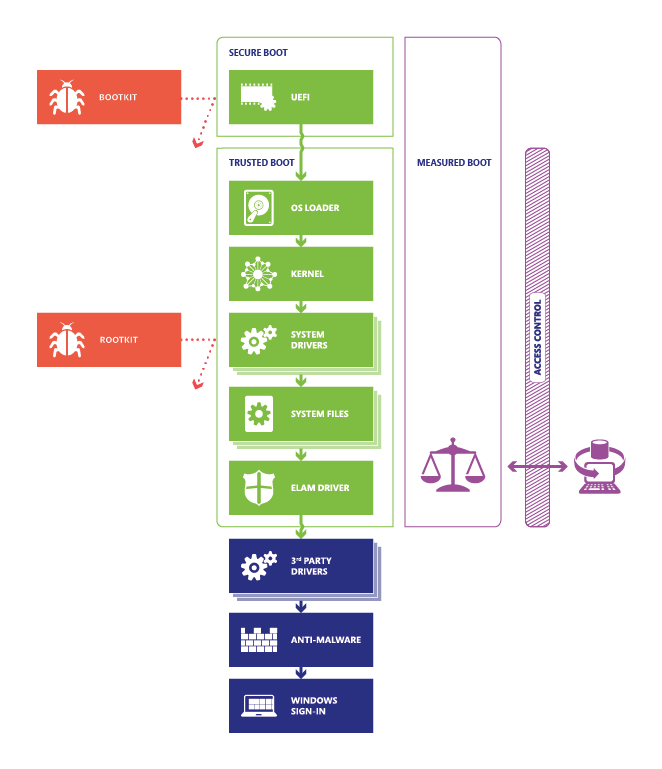
This process identifies if the computer has multiple OS installed in the computer. If yes, the process offers a menu with the names of the OS to choose from. Selecting windows will start the Winload.exe to boot Windows 10.

**Windows OS Loader**

Winload.exe loads the important drivers to start the Windows Kernel. The kernel establishes a connection with the hardware through the drivers and moves to the next process.

**Windows NT OS Kernel**

The kernel loads the system registry into the memory and loads the other drivers, marked as BOOT\_START. The control is then passed to the system manager process, Smss.exe. It loads the UI, the rest of the hardware and software.

**Securing the Windows 10 Boot Process**

Windows 10 have several features to help protect the system from malware. During the booting process, a malware known as rootkits can attack the system. Rootkits can bypass local logins, record keystrokes, record passwords and so on. These malwares can bypass the system during booting process.Windows 10 support 4 features to prevent rootkits:

**1. Secure Boot.** Computers with UEFI firmware and Trusted Platform Module (TPM) are configured to load only trusted OS bootloaders.

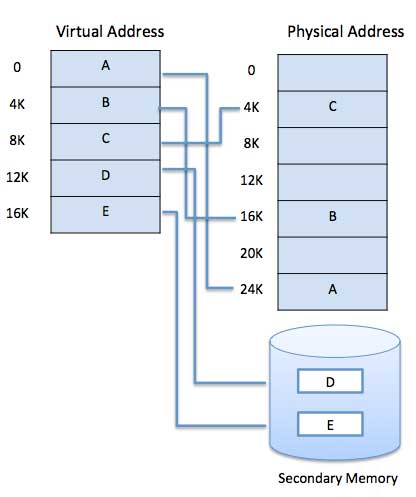
**2. Trusted Boot.** Windows check and verify every component of the startup process.

**3. Early Launch Anti**-**Malware (ELAM).** Verifies all drivers before they load, and prevents unapproved drivers from loading.

**4. Measured Boot.** The firmware logs the boot process. Windows can send the report to a trusted server, that can assess the health of the computer.

# **Virtual Memory Management**

In Windows 10, the memory manager implements virtual memory to provide services such as mapping files, large memory support, cache memory support and so on.

**Virtual Address Space**

Each process has its own virtual address space. Threads cannot access memory space of another process, protecting a process from being corrupted by other processes. Virtual address space is an internal data structure, which corresponds to their actual physical address.

The processes in the physical memory has a set of pages in the virtual memory, known as working set. The pages in the working set are shared between these processes. Removing a page from a working set of a process will not affect other pages. Removing a page from the working set of all processes, the page becomes a transition page. The transition page is then cached in the memory until it is referred to again, or repurposed.

The memory manager creates memory pools, that the system uses to allocate memory. There are two types of memory pools, nonpaged pool and paged pool. These memory pools are mapped into the virtual address space of each process, and are reserved for the system. Kernel objects allocated, are used in the nonpaged pool. Paged pools can be paged in and out of the system, depending on the circumstances. Single processors have 3 paged pool, while multiprocessors have 5 paged pools in Windows 10.

**Virtual Memory Functions**

Virtual Memory functions allow processes to manipulate pages in the virtual address space. They can perform 8 operations.

1. Reserve a range of a process’s virtual address space.
2. Commit a range of reserved pages.
3. Specify read/write, read only or no access for a range of committed pages.
4. Free range of reserved pages.
5. Decommit a range of committed pages.
6. Lock pages of committed pages into physical memory.
7. Obtain information about pages.
8. Change the access protection for a specified range of committed pages.

# **Extra: Dynamic Device Support**

1. Early in the history of the PC industry, computer configurations were fairly static, although new devices might occasionally be plugged into the serial, printer, or game ports on the back of a computer. The next steps toward dynamic configuration of PCs were laptop docks and PCMCIA cards. Using such a device, a PC could quickly be connected to or disconnected from a full set of peripherals. Contemporary PCs are designed to enable users to plug and unplug a huge host of peripherals frequently.
2. Support for dynamic configuration of devices is continually evolving in Windows. The system can automatically recognize devices when they are plugged in and can find, install, and load the appropriate drivers often without user intervention. When devices are unplugged, the drivers automatically unload, and system execution continues without disrupting other software. Additionally, Windows Update permits downloading of third-party drivers directly through Microsoft, avoiding the usage of installation DVDs or having the user scour the manufacturer’s website.
3. Beyond peripherals, Windows Server also supports dynamic hot-add and hot-replace of CPUs and RAM, as well as dynamic hot-remove of RAM. These features allow the components to be added, replaced, or removed without system interruption. While of limited use in physical servers, this technology is key to dynamic scalability in cloud computing, especially in Infrastructure as- a-Service (IaaS) and cloud computing environments. In these scenarios, a physical machine can be configured to support a limited number of its processors based on a service fee, which can then be dynamically upgraded, without requiring a reboot, through a compatible hypervisor such as Hyper-V and a simple slider in the owner’s user interface.

# **Citations :**

[1] http://www.microsoft.com

[2] <http://www.in.techradar.com>

[3] https://blogs.windows.com/windowsexperience/2015/05/13/introducing-windows-10-editions

[4] https://en.wikipedia.org/wiki/Windows\_10

[5] https://www.researchgate.net/publication/284484799\_Windows\_10/link/565401c908aeafc2aabb6434/download