

|  |
| --- |
| *UCN, University College of Northern Denmark IT-Programme AP Degree in Computer Science dmaj0920* |
| Windows XP Technology Mini Project |
|  |
| Attila Bergel  Martin Bujňák  Marek Ferdinand  Dominik Nicolas Klasić  Adam Zacios  30.03.2021-26.04.2021 |

# Abstract/Resume

This Mini project was assigned to us by our Technology professor Karsten Jeppesen during our second semester at our Computer science education on UCN. Firstly before beginning the work on this project, we had to choose an operating system which we will write about.

We chose Microsoft Windows XP, as many of us have used this operating system and we felt going back to it would be really nostalgic. Our next reason for picking Windows XP is that it is considered to be one of the best Windows operating systems ever created, and we wanted to dig deeper into it to find out why it is so popular until this day.

# 

# Preface

This report’s main purpose is to give information to the reader about the Microsoft Windows XP operating system. Some of the information may include thread model, memory management, scheduling, I/O model, and any specific hardware requirements this operating system may have.

Table of contents

[1. Abstract/Resume 1](#_Toc70334931)

[2. Preface 1](#_Toc70334932)

[3. General information about the operating system 2](#_Toc70334933)

[4. Thread model 3](#_Toc70334934)

[5. Memory management 3](#_Toc70334935)

[6. Scheduling 4](#_Toc70334936)

[7. I/O model 5](#_Toc70334937)

[8. Hardware requirements 6](#_Toc70334938)

[10. Literature 9](#_Toc70334939)

# General information about the operating system

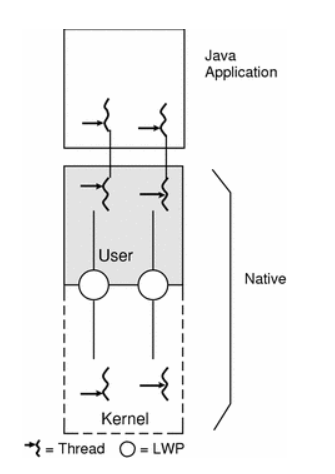
The history of the Windows operating systems is very vast, therefore only a summary is possible in this project’s scope. Since the initial release of MS-DOS in 1981, Microsoft’s operating system had unimaginable effort, time and investment put into its development, to the point that it is completely unrecognizable to the current build of their system, the Windows 10. Undoubtedly, that is also the result of 40 years of technological advances and more possibilities opening up for companies that focus on operating systems and computer software in general. The chronological order of the builds which were released after the MS-DOS is as follows: MS-DOS, Windows 1.0(W1.0), W2.0, W2.1x, W3.0, W3.1x, W95, W98, Windows ME, Windows XP, Windows Vista, W7, W8, W8.1, and finally the latest version of their operating system, Windows 10.

On account of the fact that we are focusing on Windows XP, writing about the history and development of the previous or later versions of the software will be mostly left out and kept brief in the case of an important comparison.

Windows XP is the result of 19 years of development in the Windows NT family, meaning New Technology, also possibly a play on words because of the original target processor - Intel i860 (N-Ten). On release in 2001, it received praise from many critics, noting specifically its improved performance and greater overall stability. There was some concern about the new licensing system/product activation system, however, the pros outweighed the cons in the end for the general public. It was a great success for Microsoft and resulted in a large demand for their operating systems all over the world. Presently in 2021, around 77-87% of all operating systems used are from Microsoft.

# Thread model

From our findings online, Windows XP uses a One-to-one thread model. It is among the earliest implementations of true multithreading. Each user-level thread is known to the kernel, and the kernel can be accessed by all the threads at the same time. The main problem with the one-to-one model is that it places a restriction of carefulness on the user, as each additional thread adds more “weight” to the process. Many other implementations of this model, such as Windows NT and the OS/2 threads package, limit the number of threads supported on the system.



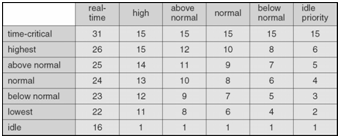
**One-to-one multithreading model**

# Memory management

The responsibility of the OS is to ensure that the physical memory is shared as efficiently as possible. Windows XP memory manager will take on a problem called memory fragmentations. It refers to the situation where the available memory becomes broken into small pieces that are not large enough to be used by the applications. The goal is to provide applications with a logical representation of memory (which is often called virtual memory). In a virtual memory system, the addresses that the applications used to access are virtual, not physical memory. The applications treat the virtual memory as though it were physical memory. Meanwhile, Windows XP OS can move code and data in physical memory whenever necessary, therefore when an application wants to access memory using a virtual address, Windows XP will translate the virtual address into a physical address.

# Scheduling

Windows XP uses a quantum-based preemptive priority scheduling algorithm. Simply put, it ensures that the highest priority thread will run. The dispatcher (a part of Windows XP which is responsible for the scheduling) uses an intricate algorithm based on a 32-level priority complex, composed of a variable class, containing levels 1-15, and a real-time class containing ranges from 16-32, however, there is further categorization in the algorithm.

A process in the Windows XP operating system can belong to several priority classes, specifically: real-time, high priority, above-normal, normal, below normal, and idle. It is worth mentioning that the variable class priorities (levels from 1-15) can change. Moreover, there is also a relative priority within each of the priority classes, namely time-critical, highest, above normal, normal, below normal, lowest, and idle. For better visualization of the priority complex, priority classes, and their respective relative priorities, a table has been included below this paragraph.

What does this thorough categorization do for the scheduling itself? It creates a system of queues for every single priority, which the dispatcher analyses from highest to lowest and tries to find a thread that is ready to run. In case that there is not a single process ready to run, an idle thread gets executed. An idle priority class is very important for applications that periodically update/monitor the system, such as screen savers, because if an idle priority class is assigned to these programs, it prevents their threads from interfering with higher priority threads.

Generally, using higher than normal priority for processes is not recommended for extended periods of time, since it could block out other lower priority, yet still necessary threads that need processor time.

# I/O model

The Windows XP I/O model is a very vast and difficult topic, so for this report, we decided to do just a small and easier abstract. The Windows kernel-mode I/O manager manages the communication between applications and the interfaces provided by device drivers. Because devices may operate at speeds that do not match the operating system’s speed, the communication between the device drivers and the operating system is primarily done through I/O request packets (IRPs). These packets are very similar to Windows message packets or network packets. They are passed from the operating system to specific drivers and from one driver to another.

The Windows I/O system provides a layered driver model, also called stacks. Typically IRPs go from one driver to another in the same stack to facilitate communication. For example, a joystick driver would need to communicate to a USB hub, which in turn would need to communicate to a USB host controller, which would then need to communicate to the rest of the computer hardware through a PCI bus. The stack consists of a joystick driver, USB hub, USB host controller, and the PCI bus. This communication is coordinated by having each and every driver in the stack receive and send IRPs.

It cannot be stressed enough that the driver must send and receive IRPs on a timely basis for the whole stack to operate efficiently and correctly. If your driver is part of a stack and doesn’t properly receive, handle, and pass on the information, your driver may cause system crashes.

# Hardware requirements

Here is a list of hardware requirements by each edition of the Windows XP operating system.

|  |  |  |
| --- | --- | --- |
| **Windows XP Home/Professional Edition** | | |
|  | Minimum | Recommended |
| CPU | Pentium or compatible, 233 MHz  -BIOS or compatible firmware | Pentium or compatible, 300 MHz  -BIOS or compatible firmware |
| Memory | 64MB | 128MB |
| Free space | 1.5 GB  -Master boot record used | * +661 MB for Service Pack 1 and 1a * +1.8 GB for Service Pack 2 * +900 MB for Service Pack 3 |
| Media | CD-ROM drive or compatible | |
| Display | SUPER VGA (800x600) | |
| Sound hardware | N/A | Sound card plus speakers/headphones |
| Input devices | Keyboard, mouse | |

|  |  |  |
| --- | --- | --- |
| **Windows XP Professional x64 Edition** | | |
|  | Minimum | Recommended |
| CPU | x86-64 or compatible  -BIOS or compatible firmware | |
| Memory | 256MB | |
| Free space | 1.5GB  -Master boot record used | |
| Media | CD-ROM drive or compatible | |
| Display | SUPER VGA (800x600) | |
| Sound hardware | N/A | Sound card plus speakers/headphones |
| Input devices | Keyboard, mouse | |

|  |  |  |
| --- | --- | --- |
| **Windows XP 64-Bit Edition** | | |
|  | Minimum | Recommended |
| CPU | [Itanium](https://en.wikipedia.org/wiki/Itanium) 733 [MHz](https://en.wikipedia.org/wiki/MHz) | [Itanium](https://en.wikipedia.org/wiki/Itanium) 800 [MHz](https://en.wikipedia.org/wiki/MHz) |
| Memory | 1GB | |
| Free space | 6GB | |
| Media | CD-ROM drive or compatible | |
| Display | SUPER VGA (800x600) | |
| Input devices | Keyboard, mouse | |

1. Conclusion

This was a rather new approach for a mini project compared to the other ones since here we didn’t have to create any UML diagrams or any code. However researching was very difficult, as sources for these topics are rather scarce, especially for the I/O model and the scheduling. We chose Windows 10 as our first choice, however, we quickly changed from that version of Windows because there was close to zero information concerning its in-depth processes, and therefore we chose Windows XP instead.

An inconvenience that occurred during this project was the fact that we had no dedicated time for this project and had to work on it during our free time and during the K2/K3/K4 activities.

# Literature

* Lake, Matt (September 3, 2001). ["Windows XP"](https://web.archive.org/web/20011119231138/http://home.cnet.com/software/0-6688749-8-7007240-2.html). CNET Review. Archived from [the original](http://home.cnet.com/software/0-6688749-8-7007240-2.html) on November 19, 2001. Retrieved May 29, 2014.
* ["Desktop Windows Version Market Share Worldwide | StatCounter Global Stats"](https://gs.statcounter.com/windows-version-market-share/desktop/worldwide/#monthly-201802-202102). gs.statcounter.com. Statcounter. Retrieved March 2, 2021.
* Fadoo Engineers. Fadoo engineers [online]. 2017 [cit. 2021-4-26]. Available from: [http://www.faadooengineers.com/online-study/post/cse/operating-system/256/windows-xp-schedulin](http://www.faadooengineers.com/online-study/post/cse/operating-system/256/windows-xp-scheduling)g
* Oracle Corporation [online]. Multithreading Models 2010 [cit. 2021-4-26]. Available from: https://docs.oracle.com/cd/E19620-01/805-4031/6j3qv1oej/index.html
* https://www.coursehero.com/file/p2m17oe/OPERATING-SYSTEMS-WINDOWS-XP-Memory-management-Window-XP-is-an-operating-system/
* Shivani Monga & Shubham Gupta. [online ]Windows XP Scheduling 2017 [cit. 2021-4-26]. Available from: https://www.slideshare.net/shivanimonga359/windows-xp-scheduling
* Mark Russinovich, David A. Solomon & Alex Ionescu. [online] WindowsInternals, Part 2, 6th Edition2012 [cit. 2021-4-26]. Available from: https://www.microsoftpressstore.com/articles/article.aspx?p=2201309
* Materials and presentations from UCN on Canvas

## 