Operating Systems

An introduction

Programming – Game Development Foundations

Last modified 26/11/15 by Richard Taylor

Contents

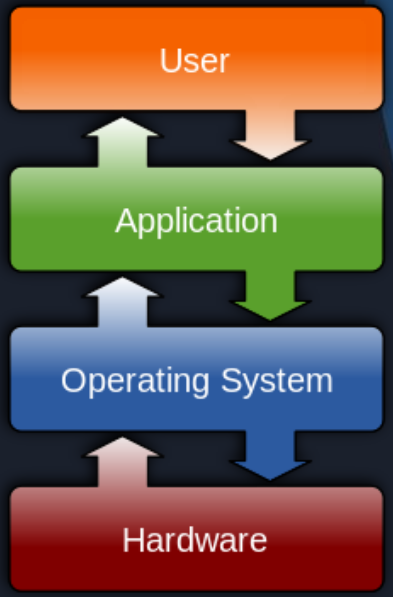
- What is an Operating System (OS)?
- Common functionality
- Comparisons, and considerations for development
- Virtual machines

Where do we find Operating Systems?

- Almost any non-trivial computer
  - Desktop and laptop PCs (Windows, OS X, Linux)
  - Phones and tablets (iOS, Android, Windows Phone)
    - Pre-”smart” phones also often had an OS!
  - Game consoles and handhelds
  - Smart watches

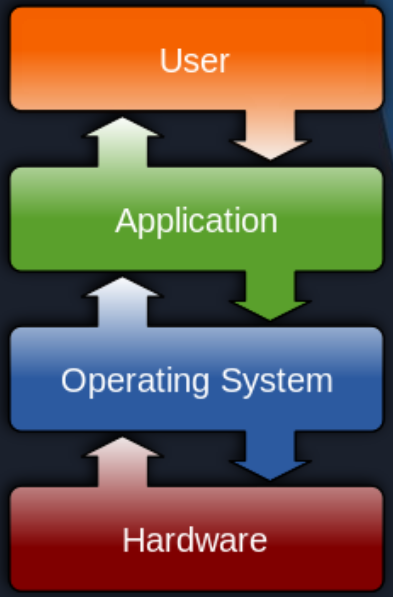
What is an Operating System?

- A layer of software that helps both users and developers work with hardware
- Provides common functionality
  - Loading and running applications!
- Enables computing tasks which would otherwise be very difficult!



What are the benefits of having an OS?

- Improved hardware support.
  - Software is written for the OS.
  - OS has “drivers” to talk to hardware.
    - “Abstraction Layer”.
  - Software developers do not have to specifically know about every possible hardware combination.



What are the benefits of having an OS?

- Ease of use.
  - Generally provides a Graphical User Interface (GUI).
  - Computer features set up once with operating system, instead of for each application.
- Resource management
  - Multiple software applications can simultaneously get input, draw graphics, play audio, accessing files or network, etc.
  - Manages multi-tasking.

What are the benefits of having an OS?

- Security
  - Not perfect by any means!
  - Example: a game you are running can’t (easily) listen for a password you’re typing into a different window.

Batch, real-time and multi-tasking

- Computers have different ways of processing data.

- These can broadly be categorised into:
  - Batch processing
  - Real-time processing
  - Multi-tasking

Batch processing

- An evolution from old, mechanical computers.

"FortranCardPROJ039.agr" by Arnold Reinhold - I took this picture of an artifact in my possession. The card was created in the late 1960s or early 1970s and has no copyright notice.. .
Licensed under CC BY-SA 2.5 via Wikimedia Commons

Bundesarchiv, B 145 Bild-F031434-0006 / Gathmann, Jens / CC-BY-SA 3.0

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Batch processing

- Batch processing is where a task or job is:
  - configured completely in advance, then
  - run to completion without user input.

- Modern computers don’t use punch cards, but some jobs are still done in a similar way.
  - Large data processing, such as payroll or interest calculations.
  - CG effects rendering for film.
  - Automated system updates or backups.

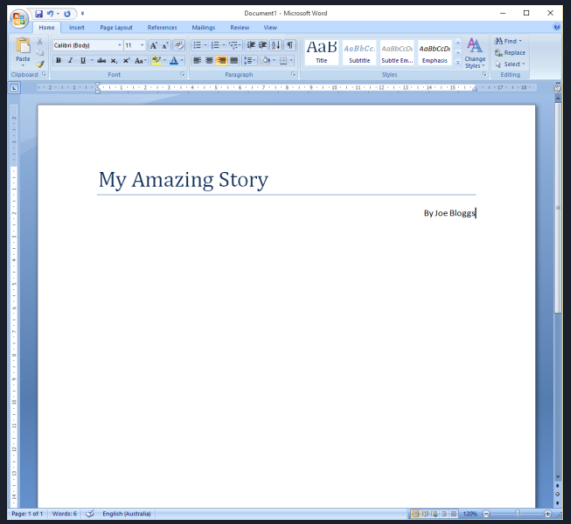
Batch processing

- Benefits include:
  - Scheduling tasks during user inactivity.
  - Scheduling repeated tasks on a regular basis.
  - Performing tasks where user interaction is undesirable.
  - Moving large tasks onto their own systems.

Real-time processing

- The software most people are familiar with uses real-time processing.

Screenshot from Fallout 4



Screenshot from Microsoft Word 2007

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Real-time processing

- Real-time processing is where:
  - processing can give near-immediate results, and
  - is performed with the potential for user interaction.

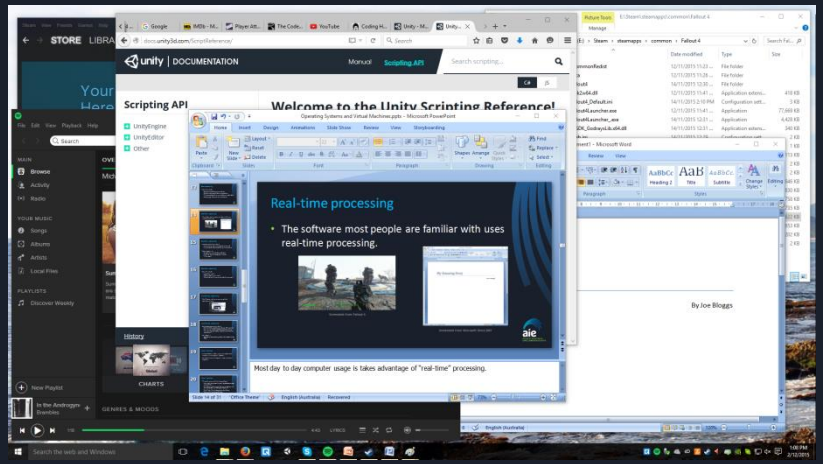
- Some examples are:
  - Word processing or desktop publishing.
  - Real-time process management software.
  - Video games!

Real-time processing

- Benefits include:
  - Immediate results.
  - Tasks can adapt to user input.
  - Allows tasks which require or are based on user input.

Multi-tasking processing

- Multi-tasking is where we can run multiple applications at once.



Screenshot of a Windows 10 desktop with a variety of apps

Multi-tasking processing

- Multi-tasking processing is where:
  - Multiple tasks or applications can be open at the same time.
  - The user can switch between many applications freely.

- An example is:
  - Standard desktop operating systems such as Windows or Mac OS X.

Multi-tasking processing

- Benefits include:
  - Have many applications open at once.
    - Unity Editor + code editor + web browser? No problem.
  - User can work in one application while another is busy or waiting.
  - More processing in less time.
    - The OS can often manage resources such that a set of tasks running concurrently could finish in less time than if they were run one after the other.

Process Scheduler

- A CPU (or core) can only run one process at a time.
- “Time slicing”.
  - CPU time is divided into “slices”, which are then divided amongst running processes.
- This is done by the “Process Scheduler”.
  - Various policies can be used.
  - Switching frequency impacts performance.

Process Scheduler

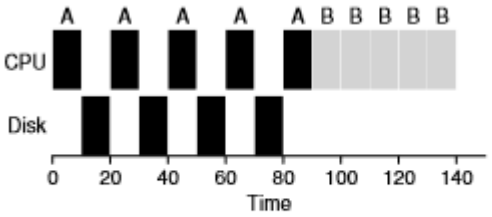


Figure 7.8: **Poor Use of Resources**

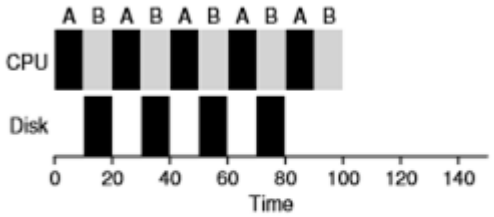


Figure 7.9: **Overlap Allows Better Use of Resources**

Arpaci-Dusseau R and A, 2015, Operating Systems: Three Easy Pieces, Chapter 7

Memory Manager

- Each process is given an “address space”.
- Instead of mapping to physical memory, processes map within their address space. This is called “virtualisation”.
- This allows operating systems to provide additional security – it stops processes from interfering with one another.
- Additionally, a process can be moved to “virtual memory”.
  - To save RAM, a process’s data can be moved to the HDD if it is inactive.
  - The Operating System then restores this to RAM when it becomes active again.

Operating System Comparison

- Different operating systems provide different capabilities and features.

- We need to take these into account when developing software.
  - Some functionality relies on OS features
  - Usability or workflow differences

Operating System Comparison

OS Feature	Windows 10	iOS 9
Multi-tasking	Unlimited*, 4 active “snapped” apps	Specific background tasks, 2 active apps, hardware dependent
File system access	Relatively open	Specified areas only
Location data access	With user permission	With user permission
Primary user input	Keyboard + pointer *or* touch screen	Touch screen

* More specifically, limited only by available resources.

Virtual Machines

- A Virtual Machine (VM) is a piece of software that emulates a complete computer system.
  - Hardware
  - Operating System and other software
- This allows us to run a computer inside of another computer.

Virtual Machines - Benefits

- Run code that you don’t trust yet. (Eg: bad OS code might corrupt data.)
- Test libraries, APIs, or logic that relies on hardware not directly available on development machines.
- Much faster than deploying to an actual device.

Virtual Machines - Benefits

- Also useful for automated testing.

- Any non-trivial software has to be tested on multiple systems!
  - Even if you're only target "Windows" you'll probably want to test on Windows 7, Windows 8 and Windows 10.
  - On the mobile front, you'll want to support and test on multiple versions of iOS and/or Android.

Virtual Machines - Limitations

- As the hardware is “emulated”, speed and compatibility are not indicative.
  - Don’t use a VM to test performance.
  - Can’t test hardware-specific features, like making phone calls.
- Licensing.
  - You still need licenses for software run on virtual machines.
- You will always want *at least* one real device to test on.

Virtual Machines

An Android Virtual Device running on a Windows 10 PC.

Summary

- Common Operating System functionality
  - Benefits for users and developers
  - Types of processing
  - Comparing operating systems
- Considerations when making and testing software
- Virtual machines

References

- Arpaci-Dusseau R and A, 2015, *Operating Systems: Three Easy Pieces (v0.90)*, various chapters
  - http://pages.cs.wisc.edu/~remzi/OSTEP/
- *iOS Developer Library*, Apple Inc., accessed 9/12/2015
  - https://developer.apple.com/library/ios/
- *Introduction to Android*, Google Inc., accessed 9/12/2015
  - https://developer.android.com/guide/index.html