Introduction

Johan Montelius

KTH

2019
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system
Choosing an operating system

What is important when choosing an operating system?
Choosing an operating system

What is important when choosing an operating system?

- What everyone else is using.
Choosing an operating system

What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
- Hardware
Choosing an operating system

What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
- Hardware
- Cost
Choosing an operating system

What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
- Hardware
- Cost
- Availability of programs
Choosing an operating system

What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
- Hardware
- Cost
- Availability of programs
Choosing an operating system

What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
- Hardware
- Cost
- Availability of programs
What is important when choosing an operating system?

- What everyone else is using.
- Look and feel, ease of use.
- Hardware
- Cost
- Availability of programs
- How it works under the hood.
What is an operating system?

Abstraction, virtualisation and managing of resource.

Abstraction

How do we create an abstraction layer that provides an environment for programming of a process?

Virtualisation

How do we create the image of dedicated hardware while in fact we have several processes sharing the same hardware?

Resource management

Given that we have limited amount of resources, how do we share them in a fair way?
What is an operating system?

Abstraction, virtualisation and managing of resource.
What is an operating system?

Abstraction, virtualisation and managing of resource.
What is an operating system?

Abstraction, virtualisation and managing of resource.

- Abstraction
What is an operating system?

Abstraction, virtualisation and managing of resource.

- Abstraction
  - How do we create an abstraction layer that provides an environment for programming of a process?
What is an operating system?

Abstraction, virtualisation and managing of resource.

- **Abstraction**
  - How do we create an abstraction layer that provides an environment for programming of a process?

- **Virtualisation**
  - How do we create the image of dedicated hardware while in fact we have several process sharing the same hardware?
What is an operating system?

Abstraction, virtualisation and managing of resource.

- **Abstraction**
  - How do we create an abstraction layer that provides an environment for programming of a process?

- **Virtualisation**
  - How do we create the image of dedicated hardware while in fact we have several process sharing the same hardware?

- **Resource management**
  - Given that we have limited amount of resources, how do we share them in a fair way?
Abstraction

The Operating System
Abstraction

A well structured world

The Operating System
Abstraction

A well structured world

a clean interface

The Operating System
Abstraction

A well structured world

a clean interface

The Operating System

Hardware
Abstraction

A well structured world

a clean interface

The Operating System

a complete mess

Hardware
Abstraction

Hardware: CPU, RAM, HD, SSD, NIC, USB, ...
Abstraction

Hardware: CPU, RAM, HD, SSD, NIC, USB....
x86_64 architecture

Hardware: CPU, RAM, HD, SSD, NIC, USB....
Abstraction

\textbf{x86\_64 architecture}

\textbf{Hardware: CPU, RAM, HD, SSD, NIC, USB...}

\begin{itemize}
  \item kernel
  \item libraries
\end{itemize}
Abstraction

x86_64 architecture

Hardware: CPU, RAM, HD, SSD, NIC, USB, ....
Abstraction

POSIX (Unix, Linux, ...)

- kernel
- libraries

×86_64 architecture

Hardware: CPU, RAM, HD, SSD, NIC, USB, ....
Abstraction

POSIX (Unix, Linux, ...)

kernel

libraries

x86_64 architecture

Hardware: CPU, RAM, HD, SSD, NIC, USB, ...
Operating system API
Operating system API

- process handling: fork, exec, wait, ...

- file operations: fopen, fclose, fread, fwrite, ...

- memory allocation: malloc, free, ...

- signal handling: signal, raise, kill, ...

- network handling: socket, listen, accept, ...

- thread handling: pthread_create, ...

- managing directory and file ownership
Operating system API

- process handling: fork, exec, wait, ...
- process communication: pipes, ..
Operating system API

- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...

POSIX: Portale Operating System Interface
Operating system API

- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
POSIX: Portale Operating System Interface

Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
Operating system API

- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...
- ...
POSIX: Portale Operating System Interface

Operating system API

- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)

- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
- ...

Command Line Interpreter

- shell: the text based interface
- scripting languages
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
- ...

POSIX: Portale Operating System Interface
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
- ...

Command Line Interpreter

shell: the text based interface
scripting languages
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
- ...

Command Line Interpreter
- shell: the text based interface
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
- ...

Command Line Interpreter
- shell: the text based interface
- scripting languages
Operating system API
- process handling: fork, exec, wait, ...
- process communication: pipes, ..
- threads handling: pthread_create, ...
- managing directory and file ownership
- network handling: socket, listen, accept, ...
- ...

The C Standard Library (ISO C18)
- memory allocation: malloc, free, ...
- signal handling: signal, raise, kill, ..
- file operations: fopen, fclose, fread, fwrite, ....
- ...

Command Line Interpreter
- shell: the text based interface
- scripting languages
- ...
int counter = 0;

void hello(char *name){
    printf("Hello: %s, %d\n", name, counter);
}

int main() {
    char *me = argv[1];
    while(counter != 10) {
        counter++;
        hello(me);
        sleep(1);
    }
    return 0;
}
Operating System

Hardware : CPU, 8GB RAM, ....
Virtualization

A: 2 GB RAM

Operating System

Hardware : CPU, 8GB RAM, ....
Virtualization

Operating System

Hardware : CPU, 8GB RAM, ....

A: 2 GB RAM
B: 2 GB RAM
<table>
<thead>
<tr>
<th>A: 4 GB RAM</th>
<th>B: 4 GB RAM</th>
</tr>
</thead>
</table>

**Operating System**

**Hardware** :  
CPU, 8GB RAM, ....
Virtualization

A: 4 GB RAM
B: 4 GB RAM
C: 32 GB RAM

Operating System

Hardware : CPU, 8 GB RAM, ....
Virtualization

Hypervisor

Hardware : CPU, 8GB RAM, ....
Virtualization

OS: Linux

Hypervisor

Hardware: CPU, 8GB RAM, ....
Virtualization

OS: Linux

OS: OSX

Hypervisor

Hardware : CPU, 8GB RAM, ....
Virtualization

OS: Linux

OS: OSX

OS: Windows

Hypervisor

Hardware: CPU, 8GB RAM, ....
Resource management

- Time: scheduling, how do we divide the execution time among processes
- Memory: efficient allocation and deallocation, malloc/free...
- Storage: HDD, SSD, ....
Time: scheduling, how do we divide the execution time among processes
Resource management

- Time: scheduling, how do we divide the execution time among processes
- Memory: efficient allocation and deallocation, malloc/free...
Resource management

- Time: scheduling, how do we divide the execution time among processes
- Memory: efficient allocation and deallocation, malloc/free...
- Storage: HDD, SSD, ....
to implement an operating system

Why is it hard to implement an operating system?
Why is it hard to implement an operating system?
Start programming today.