Introduction to OS

An Operating System (OS) is just a program, often called the *kernel* or *monitor*

Introduction to OS

An Operating System (OS) is just a program, often called the *kernel* or *monitor*

Its purpose is to

Introduction to OS

An Operating System (OS) is just a program, often called the *kernel* or *monitor*

Its purpose is to

- Manage the resources of the computer

Introduction to OS

An Operating System (OS) is just a program, often called the *kernel* or *monitor*

Its purpose is to

- Manage the resources of the computer
- Provide the applications programmer (N.B., *not* the end user of applications) with a usable programming interface to access those resources

The interface (graphical or otherwise) that the end user interacts with is *not* part of the OS

The interface (graphical or otherwise) that the end user interacts with is *not* part of the OS

That's just another program that uses the OS

The interface (graphical or otherwise) that the end user interacts with is *not* part of the OS

That's just another program that uses the OS

If the end-user sees it, it's not part of this unit!

Resources

So what are resources?

Resources

So what are resources?

- Hardware: cpu, memory, disk, network, sound, video, keyboard, mouse, printer, camera, ...

Resources

So what are resources?

- Hardware: cpu, memory, disk, network, sound, video, keyboard, mouse, printer, camera, ...
- Software: anything that controls the above, though use of the cpu (running applications) is a primary focus

Resources

Why do they need managing?

Resources

Why do they need managing?

1. They are limited with not enough to go round

Resources

Why do they need managing?

1. They are limited with not enough to go round
2. They need protection

Resources

Why do they need managing?

1. They are limited with not enough to go round
2. They need protection
3. They need to meet certain criteria

Resources

Limited?

Surely computers are so big and fast these days there is no scarcity on resources?

Resources

Limited?

Surely computers are so big and fast these days there is no scarcity on resources?

Actually, most computers are small and very limited!

Resources

Limited?

Surely computers are so big and fast these days there is no scarcity on resources?

Actually, most computers are small and very limited!

E.g., mobile phones have strict limitations on memory, cpu power and *energy consumption*

Resources

Limited?

Surely computers are so big and fast these days there is no scarcity on resources?

Actually, most computers are small and very limited!

E.g., mobile phones have strict limitations on memory, cpu power and *energy consumption*

Even big supercomputers are not yet big enough for many people

Resources

Limited?

Surely computers are so big and fast these days there is no scarcity on resources?

Actually, most computers are small and very limited!

E.g., mobile phones have strict limitations on memory, cpu power and *energy consumption*

Even big supercomputers are not yet big enough for many people

My laptop is currently running about 370 programs

Resources

Protection

Protection comes in many forms

Resources

Protection

Protection comes in many forms

- Preventing one program from accidentally (or intentionally) corrupting another program or its data: security

Resources

Protection

Protection comes in many forms

- Preventing one program from accidentally (or intentionally) corrupting another program or its data: security
- Ensuring certain resources are only available to those programs that are allowed: authorisation

Resources

Protection

Protection comes in many forms

- Preventing one program from accidentally (or intentionally) corrupting another program or its data: security
- Ensuring certain resources are only available to those programs that are allowed: authorisation
- Protecting you from your own stupid mistakes (Did you really want to delete that?)

Resources

Criteria

Popular criteria include

Resources

Criteria

Popular criteria include

- Responsiveness: making a program respond snappily or processing network packets as they arrive

Resources

Criteria

Popular criteria include

- Responsiveness: making a program respond snappily or processing network packets as they arrive
- Real Time: certain events *must* be dealt with in a (small) fixed amount of time, e.g., the controlling flaps on an airplane's wing, video streaming

Resources

Criteria

Popular criteria include

- Responsiveness: making a program respond snappily or processing network packets as they arrive
- Real Time: certain events *must* be dealt with in a (small) fixed amount of time, e.g., the controlling flaps on an airplane's wing, video streaming
- Security: prevention of accidental or malicious access or modification

Programming Interface

Another purpose of an OS is to provide an interface for the programmer:

Programming Interface

Another purpose of an OS is to provide an interface for the programmer:

The programmer who has to write applications for the machine does not want to have to know the details of the hardware: think portability (c.f. von Neumann's model)

Programming Interface

Another purpose of an OS is to provide an interface for the programmer:

The programmer who has to write applications for the machine does not want to have to know the details of the hardware: think portability (c.f. von Neumann's model)

- How do I prod this hardware to get it to do what I want?

Programming Interface

Another purpose of an OS is to provide an interface for the programmer:

The programmer who has to write applications for the machine does not want to have to know the details of the hardware: think portability (c.f. von Neumann's model)

- How do I prod this hardware to get it to do what I want?
- How do I get the best performance out of this disk, network, video?

Programming Interface

Another purpose of an OS is to provide an interface for the programmer:

The programmer who has to write applications for the machine does not want to have to know the details of the hardware: think portability (c.f. von Neumann's model)

- How do I prod this hardware to get it to do what I want?
- How do I get the best performance out of this disk, network, video?
- How should I deal with interrupts?

Programming Interface

Another purpose of an OS is to provide an interface for the programmer:

The programmer who has to write applications for the machine does not want to have to know the details of the hardware: think portability (c.f. von Neumann's model)

- How do I prod this hardware to get it to do what I want?
- How do I get the best performance out of this disk, network, video?
- How should I deal with interrupts?
- And so on

Programming Interface

We don’t want to have to re-implement everything in every program

Programming Interface

We don’t want to have to re-implement everything in every program

So the OS does this kind of thing for us

Programming Interface

We don’t want to have to re-implement everything in every program

So the OS does this kind of thing for us

Early programmers, before OSs, had to do it all themselves, for every program they wrote

Programming Interface

We don't want to have to re-implement everything in every program

So the OS does this kind of thing for us

Early programmers, before OSs, had to do it all themselves, for every program they wrote

Much better to let someone else do the hard work (a common theme in Computer Science)

Programming Interface

Having an expert do this stuff once and provide a standard interface to it for us to use is much better for us

Programming Interface

Having an expert do this stuff once and provide a standard interface to it for us to use is much better for us

- We don't have do it

Programming Interface

Having an expert do this stuff once and provide a standard interface to it for us to use is much better for us

- We don't have do it
- The expert is better at it and (presumably) understands the hardware well

Programming Interface

Having an expert do this stuff once and provide a standard interface to it for us to use is much better for us

- We don't have do it
- The expert is better at it and (presumably) understands the hardware well
- The expert is a better programmer than us and can get better performance out of the hardware

Programming Interface

Having an expert do this stuff once and provide a standard interface to it for us to use is much better for us

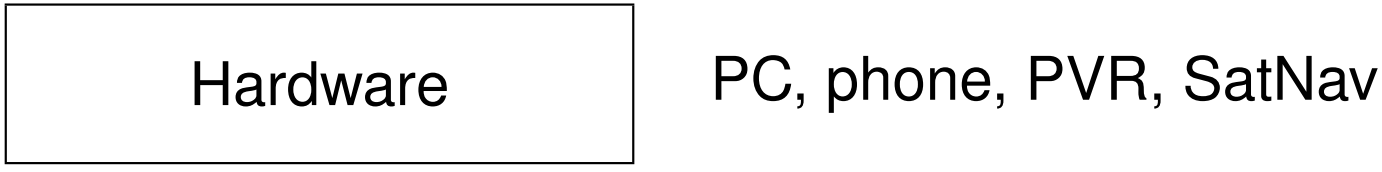
- We don't have do it
- The expert is better at it and (presumably) understands the hardware well
- The expert is a better programmer than us and can get better performance out of the hardware
- The programmer knows more Computer Science than us and knows the many pitfalls and necessary tricks that OS programming involves

Programming Interface

They do it so we don’t have to

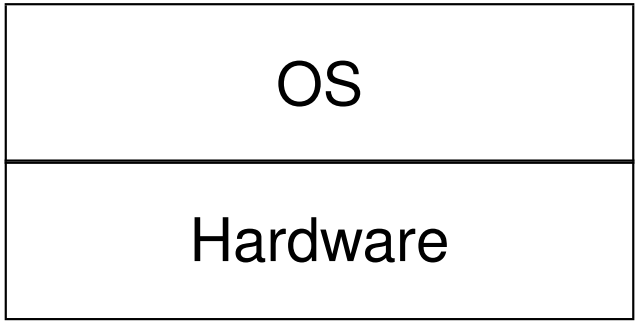
Programming Interface

Layer Abstraction:



Programming Interface

Layer Abstraction:

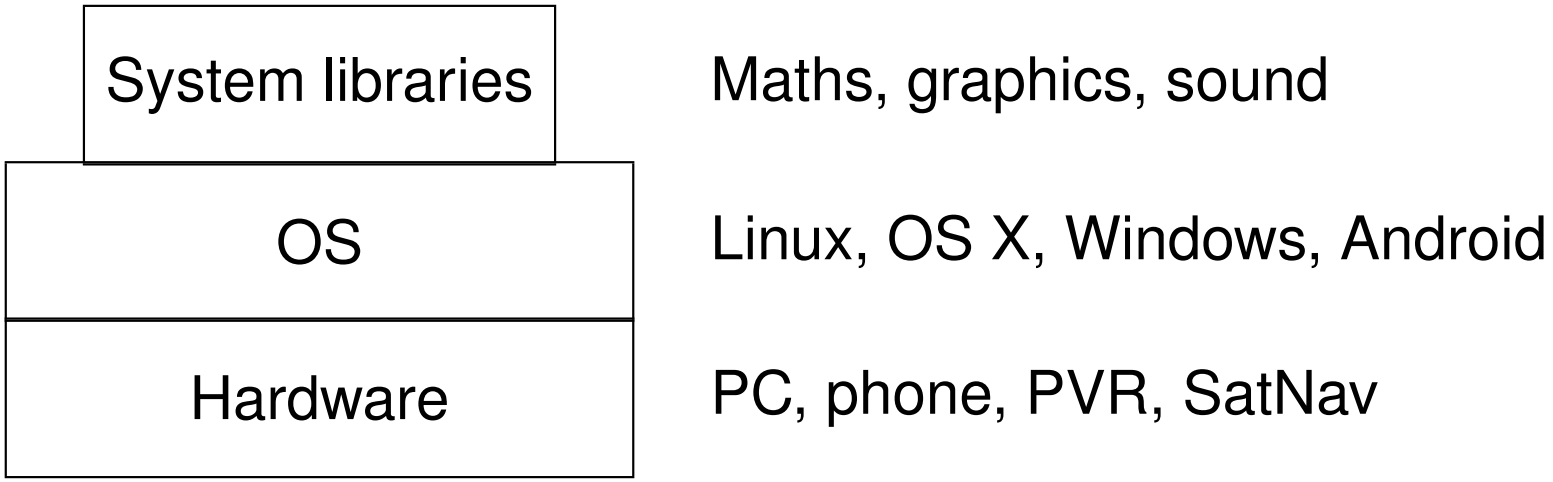


Linux, OS X, Windows, Android

PC, phone, PVR, SatNav

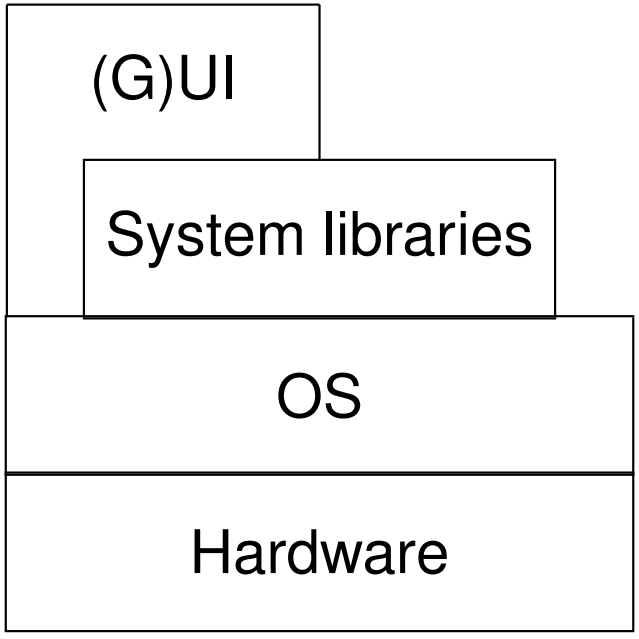
Programming Interface

Layer Abstraction:



Programming Interface

Layer Abstraction:



Command line, windowing, touch

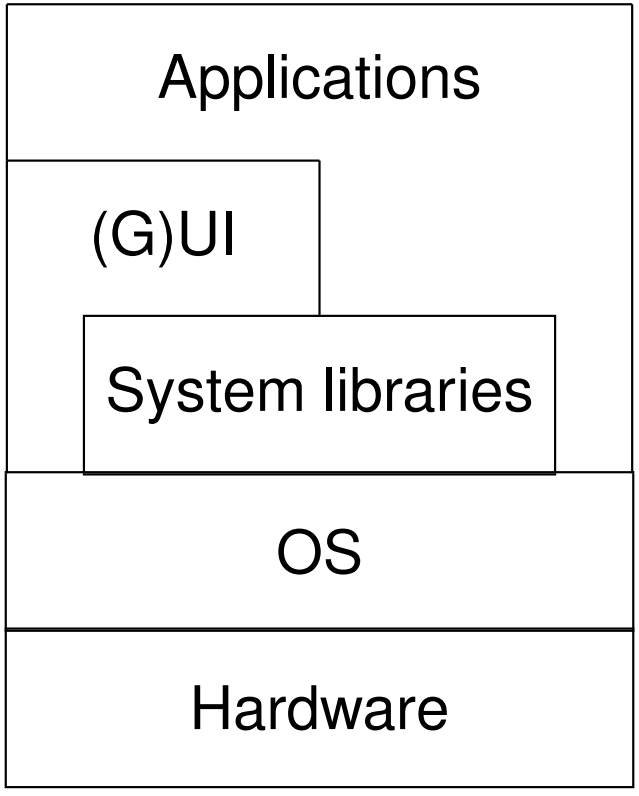
Maths, graphics, sound

Linux, OS X, Windows, Android

PC, phone, PVR, SatNav

Programming Interface

Layer Abstraction:



Browser, word processor, game

Command line, windowing, touch

Maths, graphics, sound

Linux, OS X, Windows, Android

PC, phone, PVR, SatNav

Important Point

Reemphasising a very important point:

The GUI is not part of the OS

Important Point

Reemphasising a very important point:

The GUI is not part of the OS

The GUI is just another program that uses the OS

Important Point

Reemphasising a very important point:

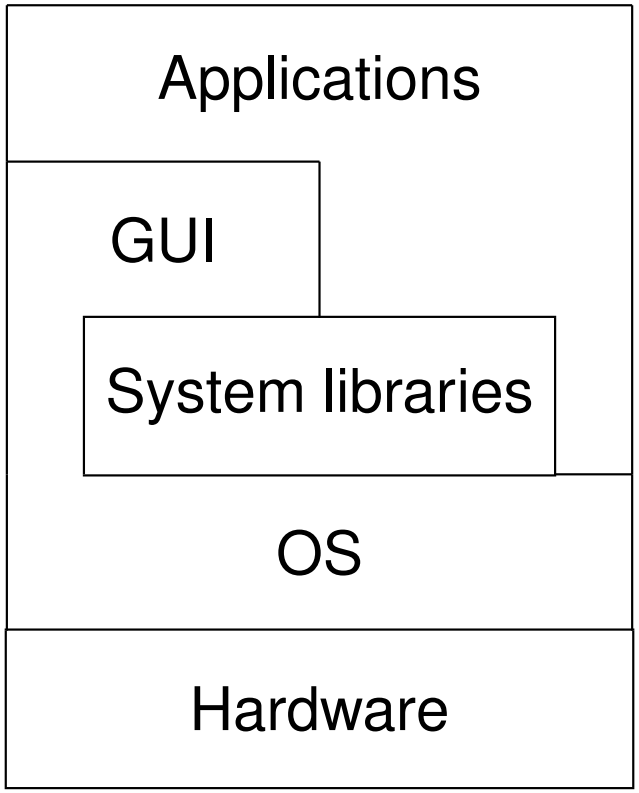
The GUI is not part of the OS

The GUI is just another program that uses the OS

There was a time when certain OS vendors tried to tie the GUI into the OS (to gain speed and commercial advantage)

Programming Interface

Bad Layer Abstraction:



Browser, word processor, game

Windows

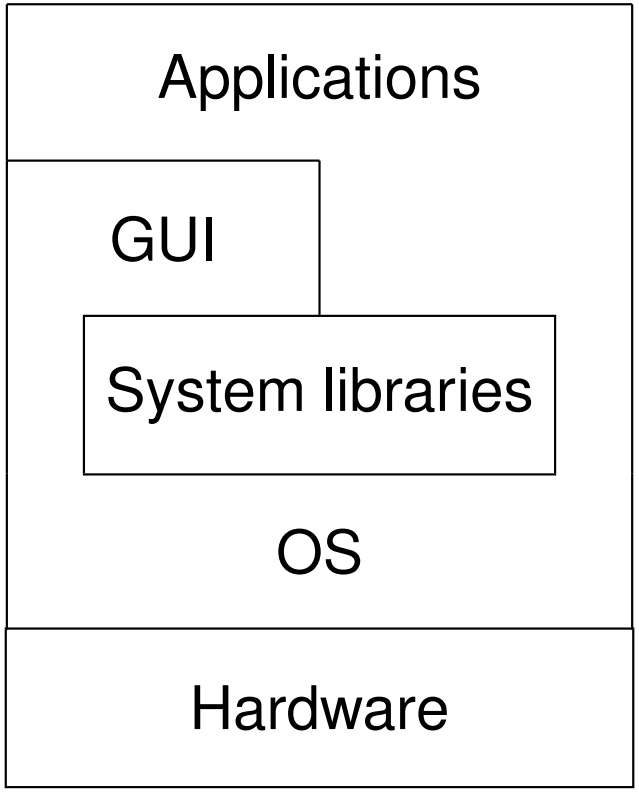
Maths, graphics, sound

Windows

PC

Programming Interface

Very Bad Layer Abstraction:



Browser, word processor, game

Windows

Maths, graphics, sound

Windows

PC

Programming Interface

This is poor design and should be avoided

Programming Interface

This is poor design and should be avoided

It was a huge source of problems: bugs in the GUI or the application would cause the OS to crash, so a poorly written program could take out the whole machine

Programming Interface

This is poor design and should be avoided

It was a huge source of problems: bugs in the GUI or the application would cause the OS to crash, so a poorly written program could take out the whole machine

It was an easy way to circumvent the security the OS provides, thus allowing attackers to access the machine

Programming Interface

This is poor design and should be avoided

It was a huge source of problems: bugs in the GUI or the application would cause the OS to crash, so a poorly written program could take out the whole machine

It was an easy way to circumvent the security the OS provides, thus allowing attackers to access the machine

An application, like a browser, could gain control of the hardware

Programming Interface

This is poor design and should be avoided

It was a huge source of problems: bugs in the GUI or the application would cause the OS to crash, so a poorly written program could take out the whole machine

It was an easy way to circumvent the security the OS provides, thus allowing attackers to access the machine

An application, like a browser, could gain control of the hardware

Fortunately things have progressed since then

Programming Interface

This is poor design and should be avoided

It was a huge source of problems: bugs in the GUI or the application would cause the OS to crash, so a poorly written program could take out the whole machine

It was an easy way to circumvent the security the OS provides, thus allowing attackers to access the machine

An application, like a browser, could gain control of the hardware

Fortunately things have progressed since then

Mostly

Another Important Point

The GUI is just a program, so it quite possible to have similar-looking GUIs running on different OSs; and different-looking GUIs running on the same OS

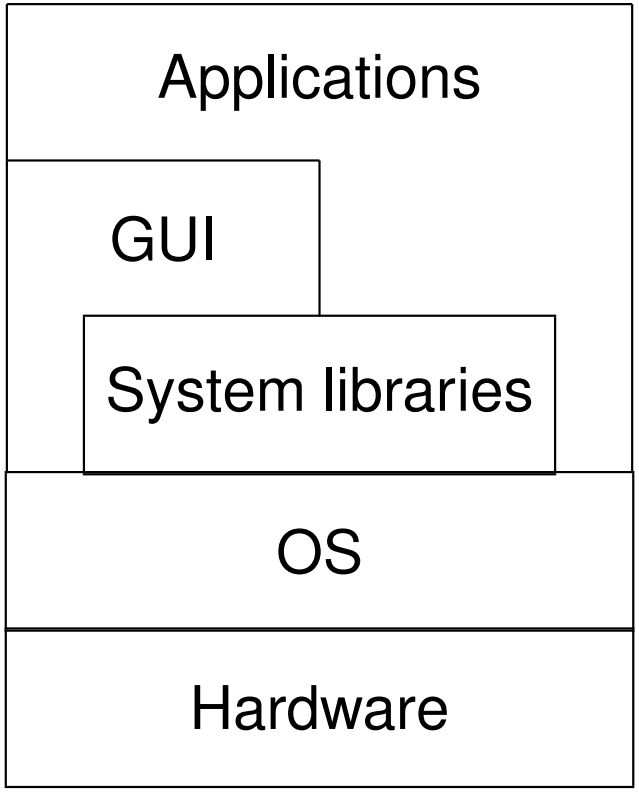
Another Important Point

The GUI is just a program, so it quite possible to have similar-looking GUIs running on different OSs; and different-looking GUIs running on the same OS

But we must be careful as some people don't realise the difference between an OS and everything else

Programming Interface

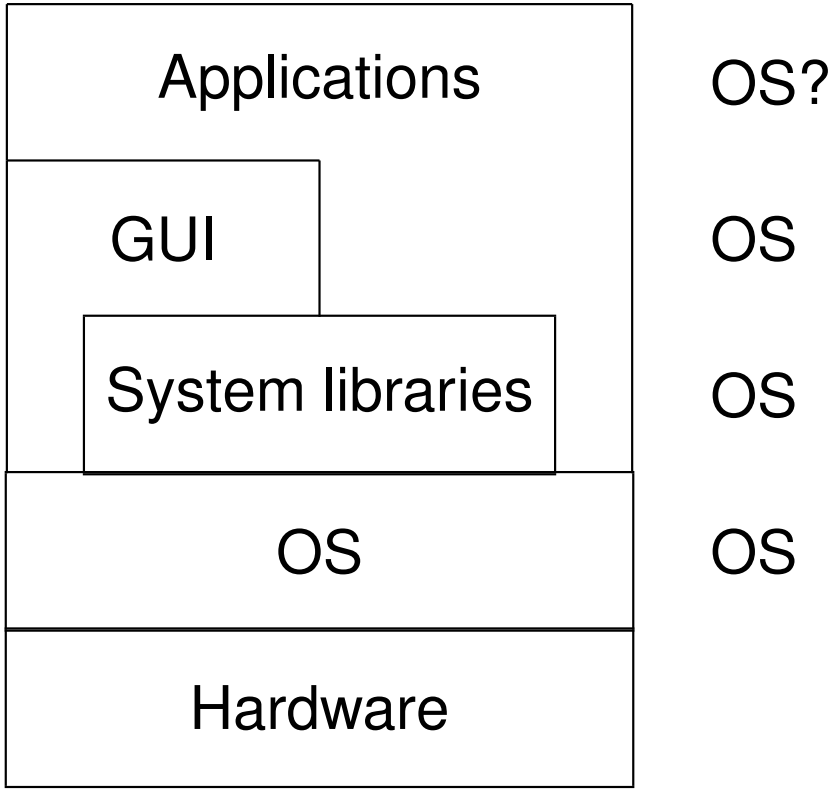
CS view:



OS

Programming Interface

Marketer's view:



One more thing

There are a couple of vital aspects of an OS that are sometimes overlooked:

One more thing

There are a couple of vital aspects of an OS that are sometimes overlooked:

1. It should be efficient and lightweight: every CPU cycle that the OS uses is one that is taken away from the user's programs

One more thing

There are a couple of vital aspects of an OS that are sometimes overlooked:

1. It should be efficient and lightweight: every CPU cycle that the OS uses is one that is taken away from the user's programs
2. It should be flexible and not get in the way of the programmer

One more thing

There are a couple of vital aspects of an OS that are sometimes overlooked:

1. It should be efficient and lightweight: every CPU cycle that the OS uses is one that is taken away from the user's programs
2. It should be flexible and not get in the way of the programmer

So a perfect OS would be completely invisible!

Background

Operating Systems have been around nearly as long as computers

Background

Operating Systems have been around nearly as long as computers

. . . but not quite as long

Background

Operating Systems have been around nearly as long as computers

. . . but not quite as long

So long, though, that some (most!) people don't distinguish between computers and their OS

Background

Operating Systems have been around nearly as long as computers

. . . but not quite as long

So long, though, that some (most!) people don't distinguish between computers and their OS

Computer Scientists ought to be more careful

Background

Operating Systems have been around nearly as long as computers

. . . but not quite as long

So long, though, that some (most!) people don't distinguish between computers and their OS

Computer Scientists ought to be more careful

So some people confuse the GUI with the OS; many confuse the GUI with the OS and the hardware!

Background

An OS is just a program so

Background

An OS is just a program so

- we can have different OSs on the same hardware

Background

An OS is just a program so

- we can have different OSs on the same hardware
- we can have the same OS on different hardware

Background

- On Intel hardware (“PC”) you can run Windows, MacOS, Linux (and lots of others)

Background

- On Intel hardware (“PC”) you can run Windows, MacOS, Linux (and lots of others)
- Until recently Mac hardware was the same as PC hardware, so you can run all of the above on a Mac — the new Mac M1 is effectively only slightly different and still run Linux

Background

- On Intel hardware (“PC”) you can run Windows, MacOS, Linux (and lots of others)
- Until recently Mac hardware was the same as PC hardware, so you can run all of the above on a Mac — the new Mac M1 is effectively only slightly different and still run Linux
- Linux runs on many kinds of hardware, including Intel, mainframes (IBM, Oracle/Sun), million processor clusters, phones (ARM), gadgets like satnavs, TVs and PVRs

Background

- On Intel hardware (“PC”) you can run Windows, MacOS, Linux (and lots of others)
- Until recently Mac hardware was the same as PC hardware, so you can run all of the above on a Mac — the new Mac M1 is effectively only slightly different and still run Linux
- Linux runs on many kinds of hardware, including Intel, mainframes (IBM, Oracle/Sun), million processor clusters, phones (ARM), gadgets like satnavs, TVs and PVRs

N.B. when we say “Intel” or “x86” we mean Intel, AMD and all the compatible architectures

Background

The understanding of the general public is such that they get the hardware and software confused

Background

The understanding of the general public is such that they get the hardware and software confused

And, more, they are usually thinking about the GUI, not the OS

Background

The understanding of the general public is such that they get the hardware and software confused

And, more, they are usually thinking about the GUI, not the OS

Some software companies encourage and make capital out of this confusion

Background

There is some reason for this confusion: Microsoft and Apple tie their GUIs indivisibly to their OSs

Background

There is some reason for this confusion: Microsoft and Apple tie their GUIs indivisibly to their OSs

Other OSs, notably Linux, allow the user a choice of many UIs and GUIs all running on the same OS kernel

Background

There is some reason for this confusion: Microsoft and Apple tie their GUIs indivisibly to their OSs

Other OSs, notably Linux, allow the user a choice of many UIs and GUIs all running on the same OS kernel

But typical users base their choices on GUIs, not OSs

Background

As applications programmers, we should choose an OS for the features that it provides:

Background

As applications programmers, we should choose an OS for the features that it provides:

- Ease of use

Background

As applications programmers, we should choose an OS for the features that it provides:

- Ease of use
- Efficiency

Background

As applications programmers, we should choose an OS for the features that it provides:

- Ease of use
- Efficiency
- Security

Background

As applications programmers, we should choose an OS for the features that it provides:

- Ease of use
- Efficiency
- Security
- Stability

Background

As applications programmers, we should choose an OS for the features that it provides:

- Ease of use
- Efficiency
- Security
- Stability
- Suitability for the task in hand

Background

As applications programmers, we should choose an OS for the features that it provides:

- Ease of use
- Efficiency
- Security
- Stability
- Suitability for the task in hand
- And so on

Background

Just a few recent operating systems:

Background

Just a few recent operating systems:

- XP/Vista/Windows 7/Windows 8/Windows 10 from Microsoft. Large, resource intensive, highly featured, previously Intel processor only, now Intel and ARM

Background

Just a few recent operating systems:

- XP/Vista/Windows 7/Windows 8/Windows 10 from Microsoft. Large, resource intensive, highly featured, previously Intel processor only, now Intel and ARM
- OS X from Apple. Large, not quite so intensive, highly featured. Based on BSD (Unix), on Intel and ARM processors (Earlier: PowerPC)

Background

Just a few recent operating systems:

- XP/Vista/Windows 7/Windows 8/Windows 10 from Microsoft. Large, resource intensive, highly featured, previously Intel processor only, now Intel and ARM
- OS X from Apple. Large, not quite so intensive, highly featured. Based on BSD (Unix), on Intel and ARM processors (Earlier: PowerPC)
- Unixes. Solaris from Sun/Oracle, IRIX from SGI, AIX from IBM, OSF/1 from DEC, etc. Large to medium.

Background

Just a few recent operating systems:

- XP/Vista/Windows 7/Windows 8/Windows 10 from Microsoft. Large, resource intensive, highly featured, previously Intel processor only, now Intel and ARM
- OS X from Apple. Large, not quite so intensive, highly featured. Based on BSD (Unix), on Intel and ARM processors (Earlier: PowerPC)
- Unixes. Solaris from Sun/Oracle, IRIX from SGI, AIX from IBM, OSF/1 from DEC, etc. Large to medium.
- Unix derivatives (reimplementations). Various BSD (including MacOS X), Linux, Hurd, etc. From small to large.

Background

Just a few recent operating systems:

- XP/Vista/Windows 7/Windows 8/Windows 10 from Microsoft. Large, resource intensive, highly featured, previously Intel processor only, now Intel and ARM
- OS X from Apple. Large, not quite so intensive, highly featured. Based on BSD (Unix), on Intel and ARM processors (Earlier: PowerPC)
- Unixes. Solaris from Sun/Oracle, IRIX from SGI, AIX from IBM, OSF/1 from DEC, etc. Large to medium.
- Unix derivatives (reimplementations). Various BSD (including MacOS X), Linux, Hurd, etc. From small to large.

Note, since the advent of smartphones, of the above OSs, Unix derivatives (Linux and Mac) are the most popular

Background

- Phones. Palm OS, Symbian, Windows CE/Mobile/Phone 7/Phone 8, Android, iPhone OS
- Experimental. Minix, Plan 9, Mach, Singularity, Amoeba, etc.
- Networking. NetWare (Novell), Cisco IOS, DD-WRT etc. For controlling networking hardware
- Distributed OSs. Management of collections of computers, or making a collection appear as a single large computer

Background

- Embedded. μCLinux, Windows Embedded, RTOS, etc. Small, resource frugal
- Real-time. QNX, μCLinux, etc. For controlling systems where a fast response is critical
- OSs for gadgets (MP3 players, etc.) Conservation of battery power is the largest problem
- Other. OS/2, MacOS 9, RISC OS, BeOS, z/OS (IBM). Various sizes

Background

- Embedded. μCLinux, Windows Embedded, RTOS, etc. Small, resource frugal
- Real-time. QNX, μCLinux, etc. For controlling systems where a fast response is critical
- OSs for gadgets (MP3 players, etc.) Conservation of battery power is the largest problem
- Other. OS/2, MacOS 9, RISC OS, BeOS, z/OS (IBM). Various sizes

Again, remember that embedded OSs outnumber PC OSs by an order of magnitude

Background

- Embedded. μCLinux, Windows Embedded, RTOS, etc. Small, resource frugal
- Real-time. QNX, μCLinux, etc. For controlling systems where a fast response is critical
- OSs for gadgets (MP3 players, etc.) Conservation of battery power is the largest problem
- Other. OS/2, MacOS 9, RISC OS, BeOS, z/OS (IBM). Various sizes

Again, remember that embedded OSs outnumber PC OSs by an order of magnitude

And we've not even mentioned historical OSs yet

Background

There are *lots* of operating systems out there, most we don't notice

Background

There are *lots* of operating systems out there, most we don't notice

The ones we do notice are failing in their purpose!

ARX project Arthur OS RISC OS AmigaOS Amiga Unix AEGIS Domain/OS vikek OS Apple DOS UCSD Pascal ProDOS GS/OS SOS Lisa OS Newton OS Mac OS 8 Mac OS 9 A/UX MkLinux Mac OS X v10.x iOS Atari DOS Atari TOS Atari MultiTOS XTS-400 BeOS Blue Eyed OS Cosmoe GCOS Burroughs MCP COS SIPROS SCOPE MACE KRONOS NOS NOS/BE RDOS AOS DG/UX CTOS DOS Deos HeartOS CP/M DR-DOS OS/8 ITS TOPS-10 WAITS TENEX TOPS-20 RSTS/E RSX-11 RT-11 VMS Domain/OS TSB Digital UNIX HP-UX Ultrix Guardian OSS OSE Towns OS Google Chrome UTX-32 INTEGRITY HDOS HT-11 HP-UX HP MIE OLERT-E Multics HeartOS DEOS iRMX ISIS-II BESYS CTSS GM OS GM-NAA I/O IBSYS IJMON SOS UMES OS/360 OS/VS SVS OS/VSn MVS/SE OS/390 z/OS DOS/360 z/VSE CP/CMS VM/370 VM/XA VM/ESA z/VM AIX/370 OpenSolaris UTS z/Linux BOS/360 MTS MUSIC/SP ORVY WYLBUR PC DOS/IBM DOS OS/2 J MultiJob GEORGE 2/3/4 TME ICL VME iVideOS LynxOS MicroC/OS-II Xenix MS-DOS Windows Singularity Midori TMX NetWare MontaVista RTXC