Sistemi Operativi e Reti di Calcolatori (SOReCa)

Corso di Laurea in *Ingegneria Informatica e Automatica (BIAR)*
Terzo Anno | Primo Semestre
A.A. 2024/2025

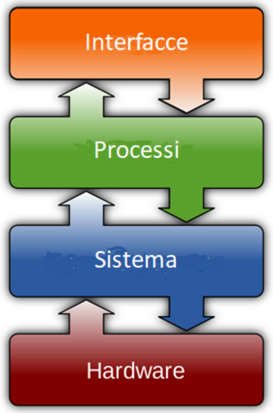
XS extra e divagazioni

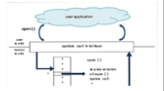
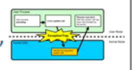
DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

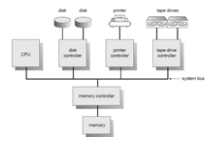
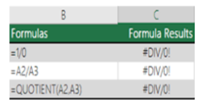
SAPIENZA UNIVERSITÀ DI ROMA

Sistemi operativi (3 CFU)

- Il sistema operativo
- Concorrenza e sincronizzazione
- Deadlock
- Inter-process communication (IPC)
- Scheduling
- Memoria centrale e virtuale
- Memoria di massa e File system
- Sicurezza informatica



Obiettivi	Funzioni	Servizi
Astrazione 	File System Sh/GUI/OLTP	Authentication/ Authorization/ Accounting
Virtualizzazione	Process Mgmt	IPC System Calls
Input/Output	Memory Mgmt Error Detection, Dual-Mode 	Boot Interrupt Handling

Lezioni: Settembre - Ottobre

WSL: Windows Subsystem for Linux

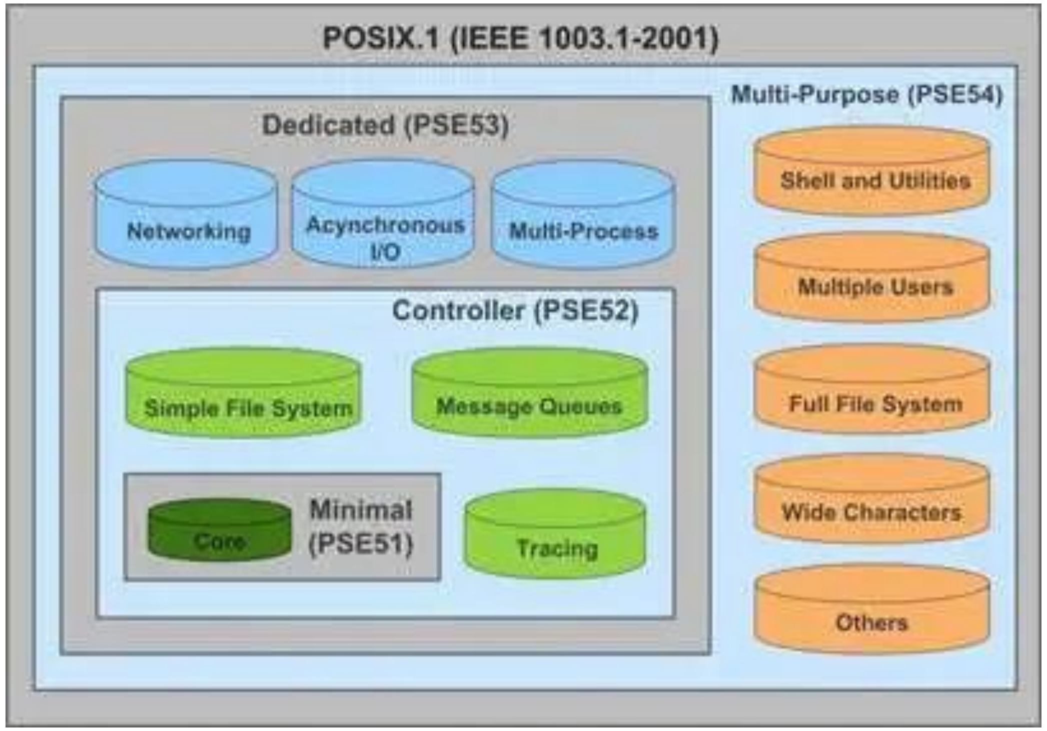
Extra: Windows System for Linux

POSIX: Portable Operating System Interface X

POSIX: family of standards specified by the IEEE Computer Society for maintaining compatibility between operating systems, mainly regarded as Unix.

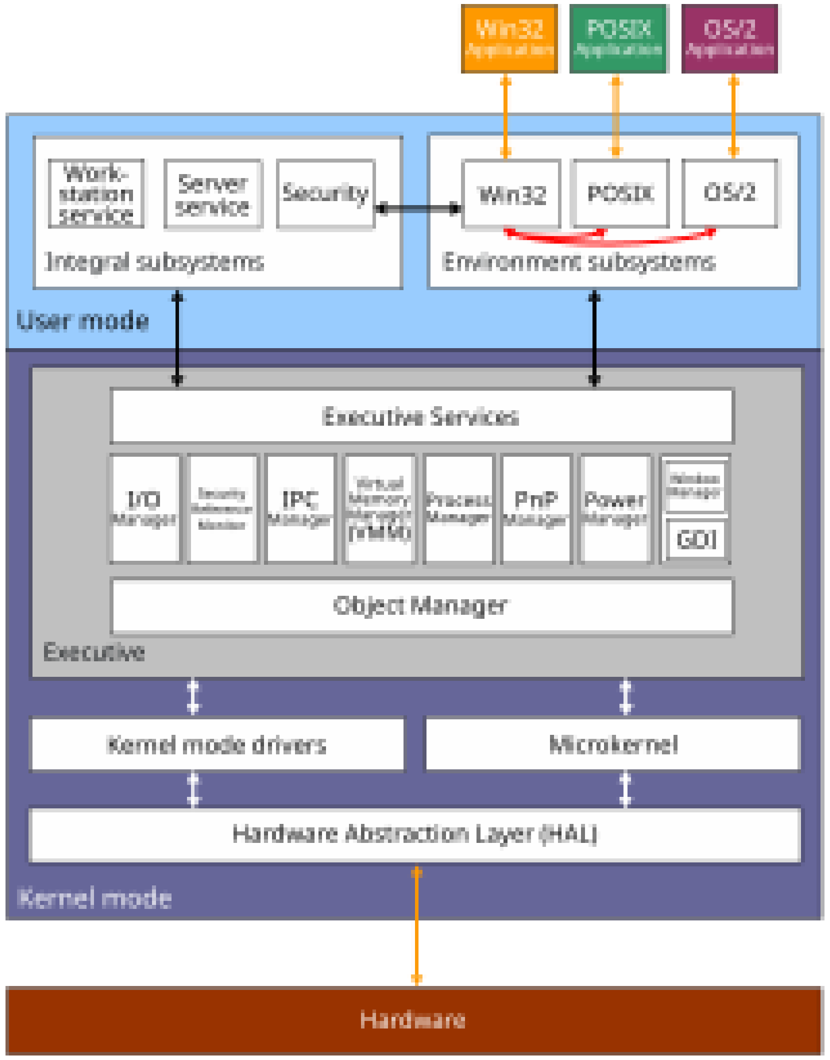
POSIX defines interfaces (command line shells, utilities etc.). There are special updates to POSIX, defining various classes of POSIX API based operating systems, like:

- PSE1: Multi-Thread Kernel
- …
- PSE52: Controller
- PSE53: multiple process networked system
- PSE54: full blown development environment
- …



Extra: Windows System for Linux

POSIX: Portable Operating System Interface X



Mac and **Linux** (embedded Linux, too) are considered to be **POSIX**, but Windows is not.

Microsoft implement only the POSIX.1 standard with the first version of Windows NT (1997) because of 1980s US federal government requirements listed in Federal Information Processing Standard (FIPS) 151-2.

Otherwise, Windows NT not have been eligible for some US government contracts.

However, Windows is not fully POSIX compliant.

The WSL overcomes this limitation by allowing developers to access a fully integrated, native Linux environment that is directly mounted to their Windows OS.

Extra: Windows System for Linux

DISM: Deployment Image Servicing and Management tool

1. **Windows Terminal**: deve essere attivato

2. **WSL**: To be activated with the command "DISM" (Deployment Image Servicing and Management tool).

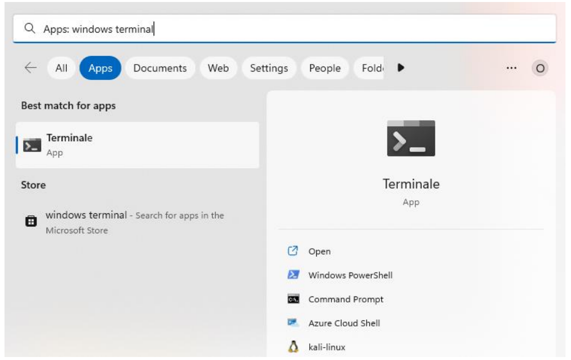
Manual installation: https://learn.microsoft.com/en-us/windows/wsl/install-manual

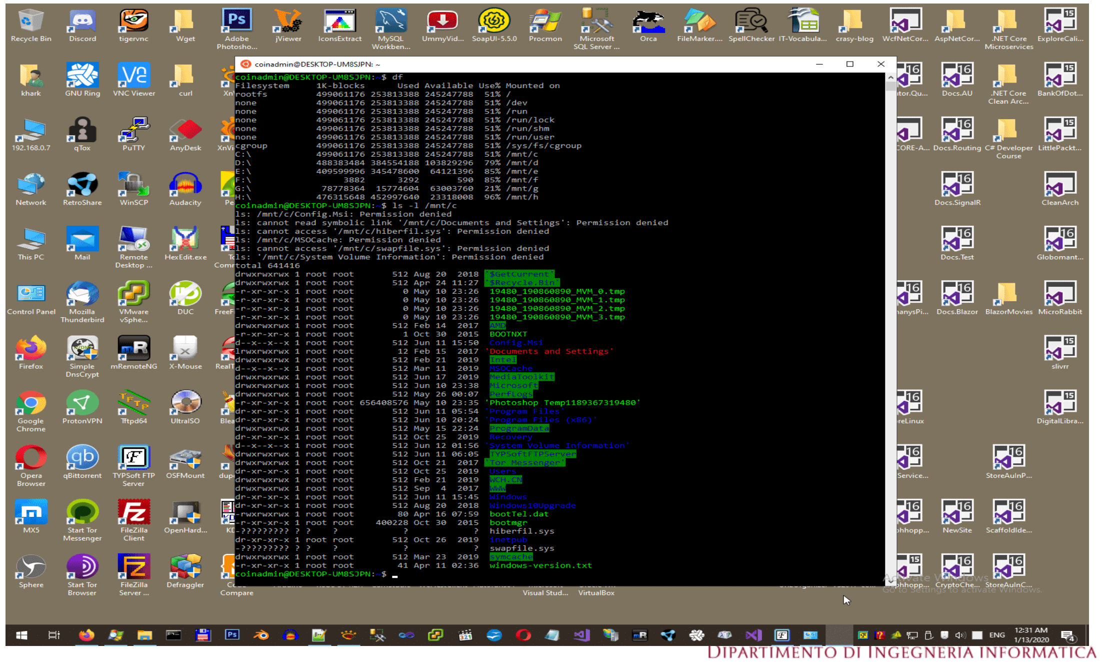
1. open powershell, running it "as an Administrator".

2. enable the WSL feature with DISM command

3. issue dism with the following parameters:

```
dism /online /enable-feature
/featurename:Microsoft-Windows-System-Linux
/all /norestart
```




coinadmin@DESKTOP-UM8SJPN: ~

```
coinadmin@DESKTOP-UM8SJPN:~$ df
Filesystem     1K-blocks      Used Available Use% Mounted on
rootfs         499061176 253813388 245247788  51% /
none           499061176 253813388 245247788  51% /dev
none           499061176 253813388 245247788  51% /run
none           499061176 253813388 245247788  51% /run/lock
none           499061176 253813388 245247788  51% /run/shm
none           499061176 253813388 245247788  51% /run/user
cgroup         499061176 253813388 245247788  51% /sys/fs/cgroup
C:\            499061176 253813388 245247788  51% /mnt/c
D:\            488383484 384554188 103829296  79% /mnt/d
E:\            409599996 345478600  64121396  85% /mnt/e
F:\                 3882      3292       590  85% /mnt/f
G:\             78778364  15774604  63003760  21% /mnt/g
H:\            476315648 452997640  23318008  96% /mnt/h
coinadmin@DESKTOP-UM8SJPN:~$ ls -l /mnt/c
ls: /mnt/c/Config.Msi: Permission denied
ls: cannot read symbolic link '/mnt/c/Documents and Settings': Permission denied
ls: cannot access '/mnt/c/hiberfil.sys': Permission denied
ls: /mnt/c/MSOCache: Permission denied
ls: cannot access '/mnt/c/swapfile.sys': Permission denied
ls: '/mnt/c/System Volume Information': Permission denied
total 641416
drwxrwxrwx 1 root root       512 Aug 20  2018 '$GetCurrent'
drwxrwxrwx 1 root root       512 Apr 24 11:27 '$Recycle.Bin'
-r-xr-xr-x 1 root root         0 May 10 23:26  19480_190860890_MVM_0.tmp
-r-xr-xr-x 1 root root         0 May 10 23:26  19480_190860890_MVM_1.tmp
-r-xr-xr-x 1 root root         0 May 10 23:26  19480_190860890_MVM_2.tmp
-r-xr-xr-x 1 root root         0 May 10 23:26  19480_190860890_MVM_3.tmp
drwxrwxrwx 1 root root       512 Feb 14  2017  AMD
-r-xr-xr-x 1 root root         1 Oct 30  2015  BOOTNXT
d--x--x--x 1 root root       512 Jun 11 15:50  Config.Msi
lrwxrwxrwx 1 root root        12 Feb 15  2017 'Documents and Settings'
drwxrwxrwx 1 root root       512 Feb 21  2019  Intel
d--x--x--x 1 root root       512 Mar 11  2019  MSOCache
drwxrwxrwx 1 root root       512 Jun 17  2019  MediaToolkit
drwxrwxrwx 1 root root       512 Jun 10 23:38  Microsoft
drwxrwxrwx 1 root root       512 May 26 00:07  PerfLogs
-r-xr-xr-x 1 root root 656408576 May 10 23:35 'Photoshop Temp1189367319480'
dr-xr-xr-x 1 root root       512 Jun 11 05:54 'Program Files'
dr-xr-xr-x 1 root root       512 Jun 10 20:24 'Program Files (x86)'
drwxrwxrwx 1 root root       512 May 15 22:24  ProgramData
dr-xr-xr-x 1 root root       512 Oct 25  2019  Recovery
d--x--x--x 1 root root       512 Jun 12 01:56 'System Volume Information'
drwxrwxrwx 1 root root       512 Jun 11 06:05  TYPSoftFTPServer
drwxrwxrwx 1 root root       512 Oct 21  2017 'Tor Messenger'
dr-xr-xr-x 1 root root       512 Oct 25  2019  Users
drwxrwxrwx 1 root root       512 Feb 21  2019  WCH.CN
drwxrwxrwx 1 root root       512 Sep  4  2017  WWW
dr-xr-xr-x 1 root root       512 Jun 11 15:45  Windows
dr-xr-xr-x 1 root root       512 Aug 20  2018  Windows10Upgrade
-rwxrwxrwx 1 root root        80 Apr 16 07:59  bootTel.dat
-r-xr-xr-x 1 root root    400228 Oct 30  2015  bootmgr
-????????? ? ?    ?            ?            ?  hiberfil.sys
dr-xr-xr-x 1 root root       512 Oct 26  2019  inetpub
-????????? ? ?    ?            ?            ?  swapfile.sys
drwxrwxrwx 1 root root       512 Mar 23  2019  symcache
-r-xr-xr-x 1 root root        41 Apr 11 02:36  windows-version.txt
coinadmin@DESKTOP-UM8SJPN:~$ _
```

Extra: Windows System for Linux

DISM: explanation

DISM: command-line tool that can be used to service and prepare Windows images.

The images include:

a) current running Windows (/online)

-> WindowsPE (repair Windows Desktop ed, "Preinstallation Environment"): https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/winpe-intro?view=windows-11

-> WindowsRE (recovery also for Server >= 2016, "Recovery Environment"): https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-recovery-environment--windows-re--technical-reference?view=windows-11

-> Windows Setup (bootable program that install the Windows O.S.): https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-setup-technical-reference?view=windows-11

b) offline Image (/image)

usage /image:<path-to-the-offline-image-file>

==> since we are configuring the current running windows --> /online

The features are functionality:

-> having a name, to be specified by the switch /featurename:

-> contained in a Package, that is "Windows Foundation Package" by default (otherwise the package should be specified with /PackageName switch).

-> enabling all parent features of the specified feature: /all

Extra: Windows System for Linux

DISM: execution

PowerShell: execute from powershell

```
PS C:\WINDOWS\system32> dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-
Linux /all /norestart


Deployment Image Servicing and Management tool

Version: 10.0.22621.1


Image Version: 10.0.22621.819


Enabling feature(s)

[==========================100.0%==========================]

The operation completed successfully.


WSL

--------------------------------------------------------------------------------
```

Now the WSL command should work.

Extra: Windows System for Linux

Preparation

Preparing the system: https://www.kali.org/docs/wsl/wsl-preparations/

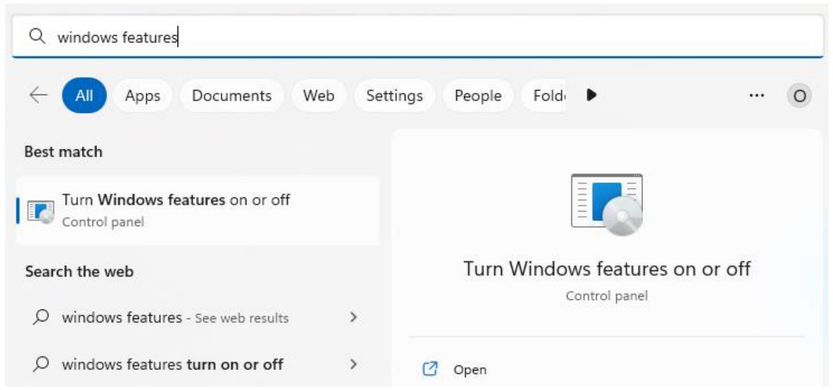
Kali Linux requires WSL2

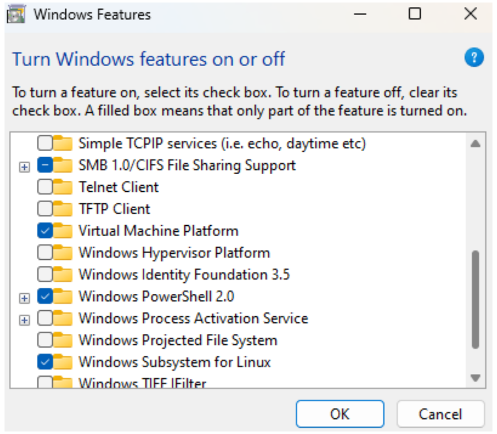
WSL update package: https://learn.microsoft.com/en-us/windows/wsl/install-manual#step-4---download-the-linux-kernel-update-package

- Step4: Download the Linux kernel update package: https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi
- Step5: Set WSL 2 as your default version

```
wsl --set-default-version 2
```

Turn On Virtual Machine Platform in Windows: https://support.microsoft.com/en-us/windows/enable-virtualization-on-windows-11-pcs-c5578302-6e43-4b4b-a449-8ced115f58e1


windows features
All
Apps
Documents
Web
Settings
People
Fold
Best match
Turn Windows features on or off
Control panel
Search the web
windows features - See web results
windows features turn on or off
Turn Windows features on or off
Control panel
Open



Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- ☐ Simple TCPIP services (i.e. echo, daytime etc)
- ■ SMB 1.0/CIFS File Sharing Support
- ☐ Telnet Client
- ☐ TFTP Client
- ✓ Virtual Machine Platform
- ☐ Windows Hypervisor Platform
- ☐ Windows Identity Foundation 3.5
- ✓ Windows PowerShell 2.0
- ☐ Windows Process Activation Service
- ☐ Windows Projected File System
- ✓ Windows Subsystem for Linux
- ☐ Windows TIFF IFilter

Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti

Sapienza Università di Roma

Extra: Windows System for Linux

Linux installation

Here in after, the wsl command could be executed from cmd.exe (PowerShell is not more strictly required).

Install: choose a standard WSL installation package (available from Microsoft Store and installable from shell command). Debian is the default

```
wsl --install -d debian
```

Add Packages: This is a minimal installation of Kali Linux, you likely want to install at least the development tools. Learn how:

- `$ sudo apt update`
- `$ sudo apt full-upgrade`
- `$ sudo apt install build-essential`
- `$ sudo apt install manpages-dev`

Mutual Exclusion Complexity

Extra: Complexity

Big O Notation

Bachmann–Landau notation or **asymptotic notation.** The letter **O** was chosen by Bachmann to stand for Ordnung, meaning the order of approximation.

Big-O notation: to **classify** algorithms according to how their run **time** or **space requirements** grow as the input size grows.

- **Constant time complexity(O(1)):** This is similar to finding something in a small drawer next to you.
- **Linear time complexity(O(n)):** It is compared to flipping through each book one by one, the more books you flip the longer it takes to complete flipping.
- **Logarithmic time complexity(O(log n))**: It is like using a structured index or catalog to narrow your search in a large library.
- **Quadratic time complexity(O(n^2))**: It is like comparing every book to every other book, the time grows as the number of books increases.
- **Cubic time complexity(O(n^3))**: Matrix multiplication and algorithms that use three nested loops are examples.
- **Exponential Time complexity (O(2^n))**: the traveling salesman problem by examining all feasible routes and determining the shortest one.

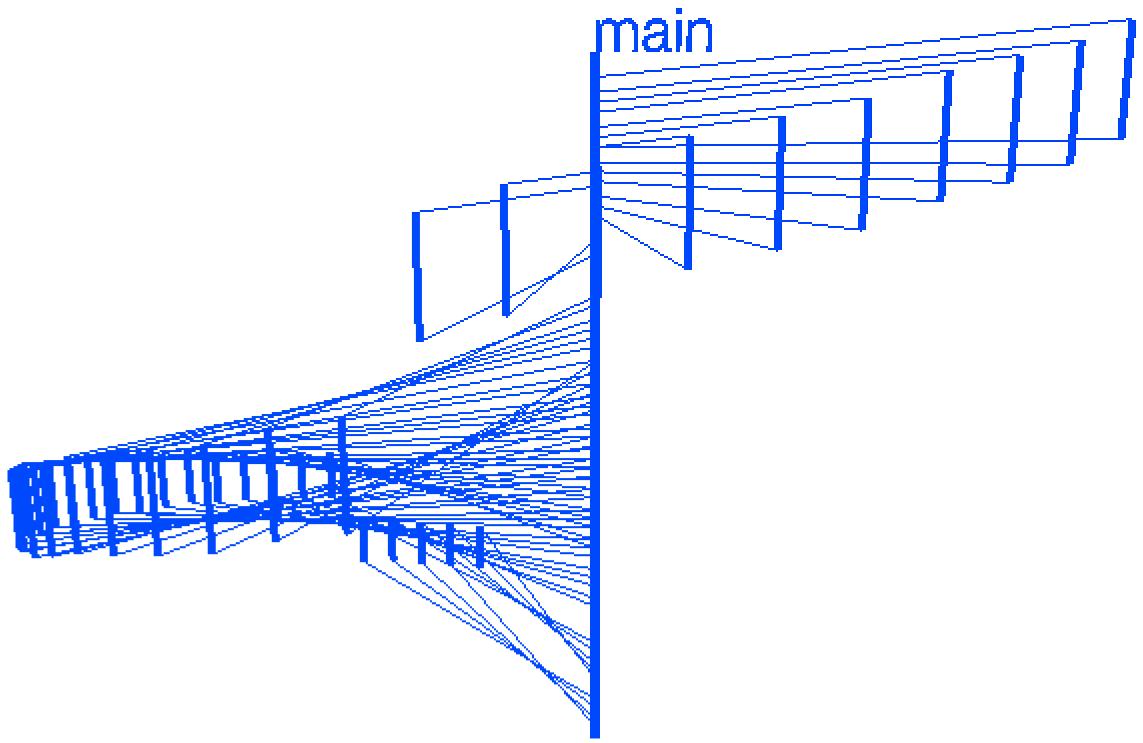
Extra: Complexity

Parallel Computation and concurrency

Typical examples of concurrent systems, like semaphores, will simply not be typable in the system for span because of linearity conditions.

Parallel computation expressed is, up to now, focused in some form of cooperative concurrency.

The analysis of complexity bounds for concurrent calculus is a challenging question to address in future.



Extra: Complexity

Examples Comparison

Some of the scenarios examinated:

Name	Relationship	#mutex	#sem	#status	Counter
Mutual Exclusion	Competition	1		1	
Sleeping Barber	Synchronization	1	2		1
Producer-Consumer: Bounded Buffer	Communication	2	2		
Reader-Writer	Communication	2	3		2
Dining Philosophers	Synchronization	1	N	N	

Where

- **Relationship**: between processes (Competition on a resource, Synchronization of Execution, Communication of Results)
- **#mutex**: number of needed binary semaphores (for mutual exclusion)
- **#sem**: number of (integer) semaphores
- **#status**: numer of (integer) variables needed for storing the status of each concurring processù
- **Counter**: numer of needed counters

WSL Compilation Erros

Lab03. es1: Makefile

Makefile

WSL: Errori di compilazione 1/3

ATTENZIONE: in WSL, la compilazione potrebbe non riuscire, generando una serie di problemi di file da includere non trovati.

```
In file included from /usr/include/features.h:392,
                 from /usr/include/errno.h:25,
                 from common.h:4,
                 from req_wrk.c:1:
/usr/include/features-time64.h:20:10: fatal error: bits/wordsize.h: No such file or
directory
   20 | #include <bits/wordsize.h>
      |          ^~~~~~~~~~~~~~~~~
compilation terminated.
```

Ciò non è dovuto alla mancanza di header file o librerie. Bensì alla richiesta implicita di usare le librerie (e, quindi, gli header file non adatti per la architettura effettiva.

Makefile

WSL: Errori di compilazione 2/3

Il Makefile riporta la seguenti indicazioni:

```
CFLAGS=-m32 -g -Wall

all: producer consumer
producer: producer.c common.h common.c
       gcc $(CFLAGS) -o producer producer.c common.c -lpthread

consumer: consumer.c common.h common.c
       gcc $(CFLAGS) -o consumer consumer.c common.c -lpthread
.PHONY: clean
clean:  rm -f producer consumer
```

} Variable Definitions

} Explicit Rules: Targets

`-m32` è una opzione del compilatore `gcc` che richiede esplicitamente di produrre codice per una architettura a 32 bit.
Poiché WSL viene eseguito all'interno di un docker, generalmente, su una architettura a 64 bit, non vi sono le infrastrutture per la simulazione di una macchina a 32. Va eliminata.

`.PHONY` è un esempio "built-in target name": il target `clean` "suona bene: va eseguito `rm .f` …" anche qualora vi fosse un file clean. Lasciare (in questo caso non serve ma è una buona abitudine inserirlo.

Makefile

WSL: Errori di compilazione 3/3

Il Makefile può essere corretto e riordinato nel seguente modo:

```
CFLAGS=-I. -lpthread -g -Wall
CC=gcc

all: producer consumer
producer: producer.c common.h common.c
        $(CC) $(CFLAGS) -o producer producer.c common.c

consumer: consumer.c common.h common.c
        $(CC) $(CFLAGS) -o consumer consumer.c common.c
.PHONY: clean
clean:  rm -f producer consumer
```

} Variable Definitions

} Explicit Rules: Targets

`-I.` indica che anche gli header file (*.h) nella directory corrente (.) possono essere inclusi anche indicandoli fra apici (<>) e non solo tramite virgolette (“”)
`-lpthread` dal momento che la libreria per I POSIX thread viene collegata ad ogni target, è opportune inserire l’opzione nei CFLAGS

Makefile

WSL: Errori di compilazione 3/3

Il Makefile può essere corretto e riordinato nel seguente modo:

```
CFLAGS=-I. -lpthread -g -Wall
CC=gcc

all: producer consumer
producer: producer.c common.h common.c
       $(CC) $(CFLAGS) -o producer producer.c common.c

consumer: consumer.c common.h common.c
       $(CC) $(CFLAGS) -o consumer consumer.c common.c
.PHONY: clean
clean:  rm -f producer consumer
```

} Variable Definitions

} Explicit Rules: Targets

`-I.` indica che anche gli header file (*.h) nella directory corrente (.) possono essere inclusi anche indicandoli fra apici (<>) e non solo tramite virgolette (“”),
`-lpthread` dal momento che la libreria per I POSIX thread viene collegata ad ogni target, è opportune inserire l’opzione nei CFLAGS.
`-g` crea l’eseguibile includendovi le informazioni utili per il debug ma senza la definizione delle macro
`-g` riporta ogni warning che dovesse risultare della compilazione

Makefile

Definizione più generale

```
CFLAGS=-I. -lpthread -g -Wall
CC=gcc
DEPS=
OBJ=server.o
USERID=123456789

-include anotherMakefile

#compiling

%.o: %.c $(DEPS)
        $(CC) -c -o $@ $< $(CFLAGS)

all: server
server: $(OBJ)
        $(CC) -o $@ $^ $(CFLAGS)

.PHONY: clean
clean:
        rm -rf *.o server *.tar.gz

dist: tarball
tarball: clean
        tar -cvzf /tmp/$(USERID).tar.gz --exclude=./.vagrant . && mv /tmp/$(USERID).tar.gz .
```

} Variable Definitions

} Include Rules

} Comment line

} Implicit Rules

} Explicit Rules: Targets

Shell

A few tips & tricks

Esecuzione in Background

$ <path-to-executable> &

```
┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./producer &
[1] 22

┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./consumer
Consumer 1 ended. Local sum is 7786
Producer 1 ended. Local sum is -754
Producer 3 ended. Local sum is -754
Producer 2 ended. Local sum is -754
Producer 0 ended. Local sum is -754
Producers have terminated. Exiting...
Consumer 0 ended. Local sum is -10802
Consumers have terminated. Exiting...
[1]+  Done                    ./producer

┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$
```

Nota: Lab03, es.2: “Consumer 0” = -“Consumer 1” + sum(“Producer i“)

Command Completion

$ <executable-first-chars> <Tab> <file1-first-chars> <Tab> ...

Command-line completion allows the user to type the **first few characters** of a command, program, or filename

```
┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./
```

then press a completion key (normally <Tab> ⭾) to see all available alternatives for filling in the rest of the items.

```
┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./
common.h     consumer     consumer.c  Makefile     producer     producer.c
┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./
```

typing the **minimum number of chars** of a command, program, or filename...

```
┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./p
```

and press a completion key (normally <Tab> ⭾) to fill in the rest of the items.

```
┌──(pottol@DESKTOP-00T2KBP)-[/mnt/c/Users/ottol/SOReCa/Labs/SOReCa.Lab03/Lab03-sol/02]
└─$ ./producer
```

Then the user normally presses Return for execution.

Redirections

stdin (0), stdout (1) stderr (2)

Just redirect (>) the output of a command, for saving the output of a command to a file,

Command	StdOut visible	StdErr visible	StdOut in File	StdErr in File	Open File
`Command > File.txt`	No	Yes	Yes	No	overwrite
`Command >> File.txt`	No	Yes	Yes	No	append
`Command 2> File.txt`	Yes	No	No	Yes	overwrite
`Command 2>> File.txt`	Yes	No	No	Yes	append
`Command &> File.txt`	No	No	Yes	Yes	overwrite
`Command &>> File.txt`	No	No	Yes	Yes	append

Command	StdOut visible	StdErr visible	Note
`Command 2>&1`	Yes	No (inside StdOut)	redirects StdErr to StdOut
`Command 1>&2`	No (inside StdErr)	Yes	redirects StdOut to StdErr

Redirections and Copy

stdin (0), stdout (1) stderr (2)

Take advantage of piping (`|`) to a specific command (`tee`), for coping the output of a command to a file,

Command	StdOut visible	StdErr visible	StdOut in File	StdErr in File	Open File
`Command	tee File.txt`	Yes	Yes	Yes	No	overwrite
`Command	tee -a File.txt`	Yes	Yes	Yes	No	append
~~…~~	~~Yes~~	~~Yes~~	~~No~~	~~Yes~~	~~overwrite~~
~~…~~	~~Yes~~	~~Yes~~	~~No~~	~~Yes~~	~~append~~
`Command |& tee File.txt`	Yes	Yes	Yes	Yes	overwrite
`Command |& tee -a File.txt`	Yes	Yes	Yes	Yes	append

OS X: Semaphores

POSIX Semaphore (tradizionali): un-named (Non supportati in OS X)

SysV Semaphore: named (supportati in OS X)

MacOS X Concepts

XNU: Kernel per Darwin Architecture

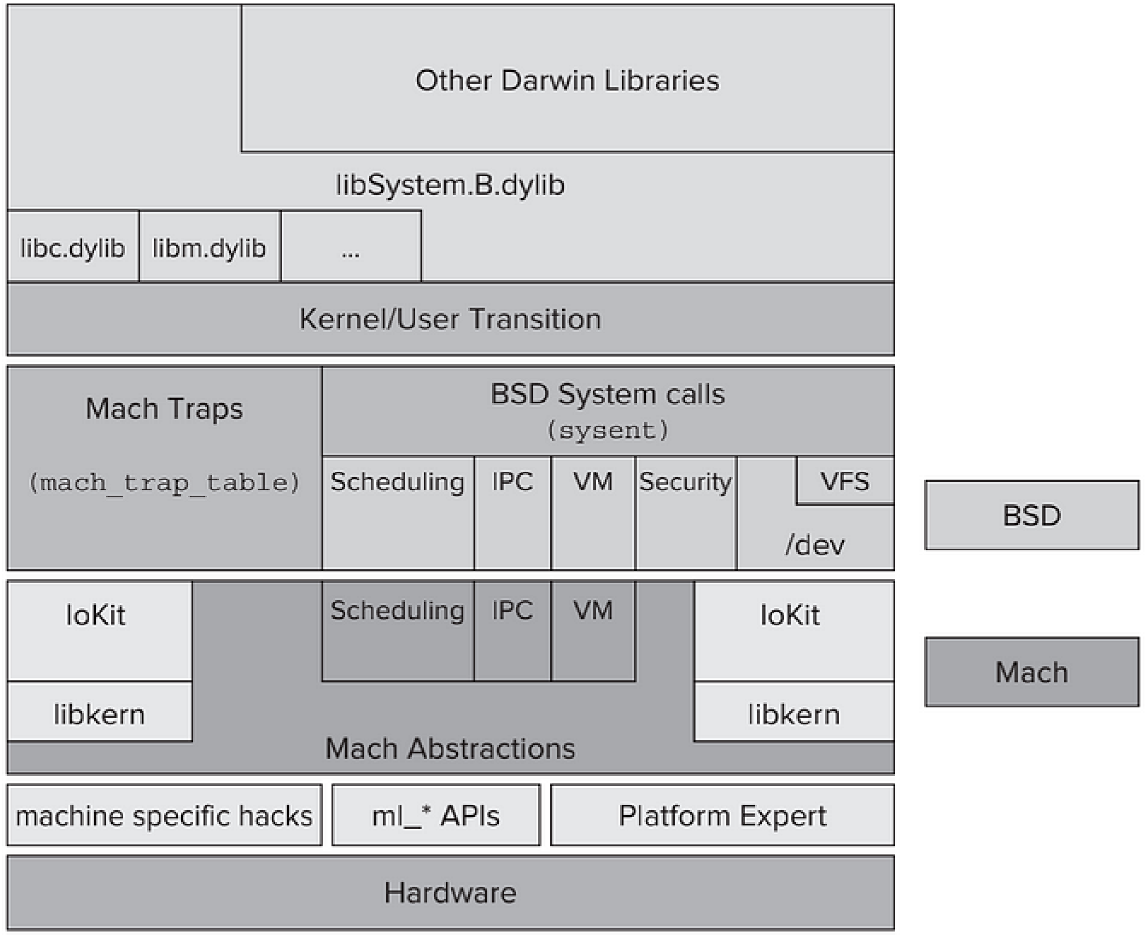
il kernel di OS X (ma anche iOS, iPadOS, WathcOS etc.) è basato su Darwin: una composizione del micro kernel Mach 3.0 ed altri contributi Open Source (principalmente, FreeBSD) per creare un singolo address space per il kernel.

Mach è un micro-kernel, sviluppato con finalità di ricerca, presso la Carnegie Mellon University, tra il 1985 ed il 1994. Il progetto è terminato con il Mach 3.0

https://www.cs.cmu.edu/afs/cs/project/mach/public/www/sources/sources_top.html

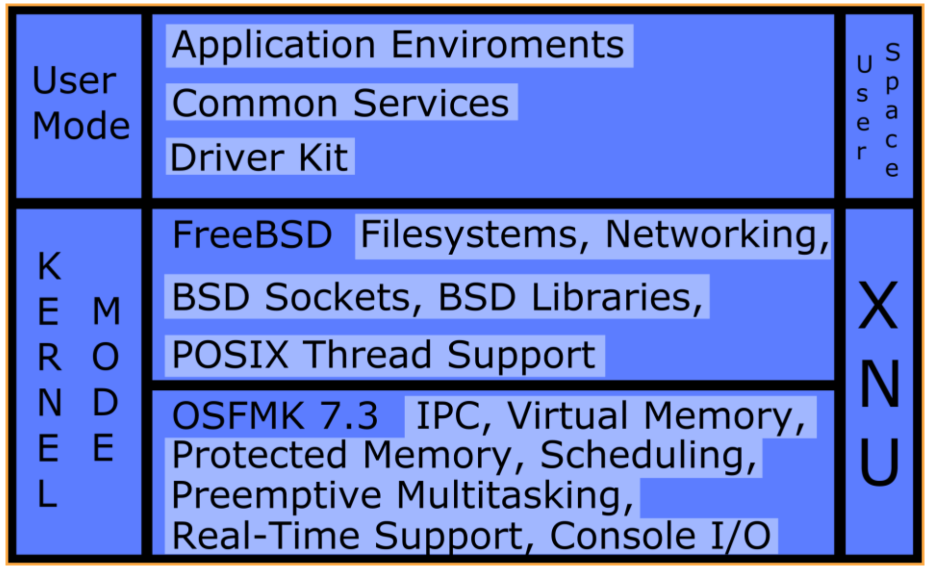
Il risultato è il **kernel ibrido XNU** (X is Not Unix), **composto** da microkernel Mach e kernel monolitico BSD:

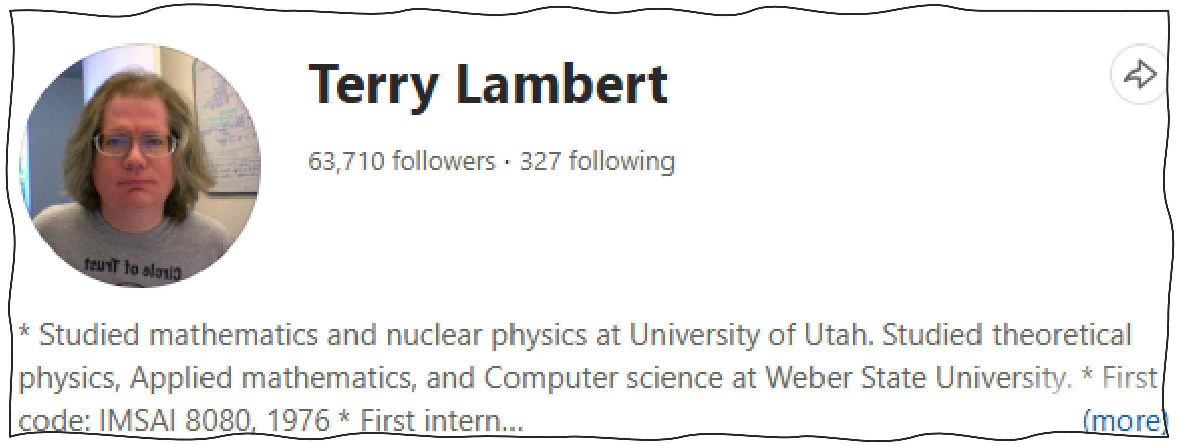
- **funzioni primitive** e servizi fondamentali → **Mach 3.0**
- **funzioni aggiuntive** e ottimizzazioni prestazionali → **BSD**
- + modifiche ed integrazioni → **Apple**



MacOS Concepts

Why does OS X not support unnamed semaphores?


User Mode
Application Enviroments
Common Services
Driver Kit
User Space
KERNEL MODE
FreeBSD Filesystems, Networking,
BSD Sockets, BSD Libraries,
POSIX Thread Support
OSFMK 7.3 IPC, Virtual Memory, Protected Memory, Scheduling, Preemptive Multitasking, Real-Time Support, Console I/O
XNU



Terry Lambert

63,710 followers · 327 following

* Studied mathematics and nuclear physics at University of Utah. Studied theoretical physics, Applied mathematics, and Computer science at Weber State University. * First code: IMSAI 8080, 1976 * First intern... (more)

Ovviamente, l'IPC proviene da Mach ed e' in corso di "ri-modulazione". Infatti, e' presente una avvertenza importante:

"The IPC facilities in OS X are in a state of transition. In early versions of the system, not all of these IPC types may be implemented. (cfr. https://developer.apple.com/library/archive/documentation/Darwin/Conceptual/KernelProgramming/Mach/Mach.html)

Gli "un-named semaphores" sono uno di queste facilities ancora in transition.

Difatti, Terry Lambert risponde cosi' alla domanda:

in sostanza, lo unnamed semaphore

- **esternamente** è trattato come int *--> 64 bit
- **internamente** è convertito e trattato come una voce nella tabella dei file aperti per processo --> int a 32 bit

➔ Per conciliare la differenza sarebbe necessaria la capacità di rappresentare i puntatori a 64 bit come interi a 32 bit.

Lui provo' ad implementare tale funzione prima di andar via da Apple ma questa interessante funzionalità non era compresa nel fork di modifiche "sostanziali". In Apple potrebbero non averla recepita ancora.