Università degli Studi di Ferrara

Corso di Laurea in Matematica - A.A. 2021 - 2022

Programmazione
Lezione 2 – Sistemi Operativi

Docente: Michele Ferrari - michele.ferrari@unife.it

Nella lezione precedente

- L’informatica moderna nasce nel corso della WW2
- Il funzionamento dei computer moderni è basato sull’architettura di von Neumann
- Il computer è un dispositivo elettronico in grado di eseguire molte attività elementari al secondo
- In un computer sono presenti più memorie in quanto la velocità di accesso alla memoria è inversamente proporzionale alla capacità della memoria stessa
- Il Software principale in un computer è il Sistema Operativo
- Un programma è un software che risolve un problema automatizzabile

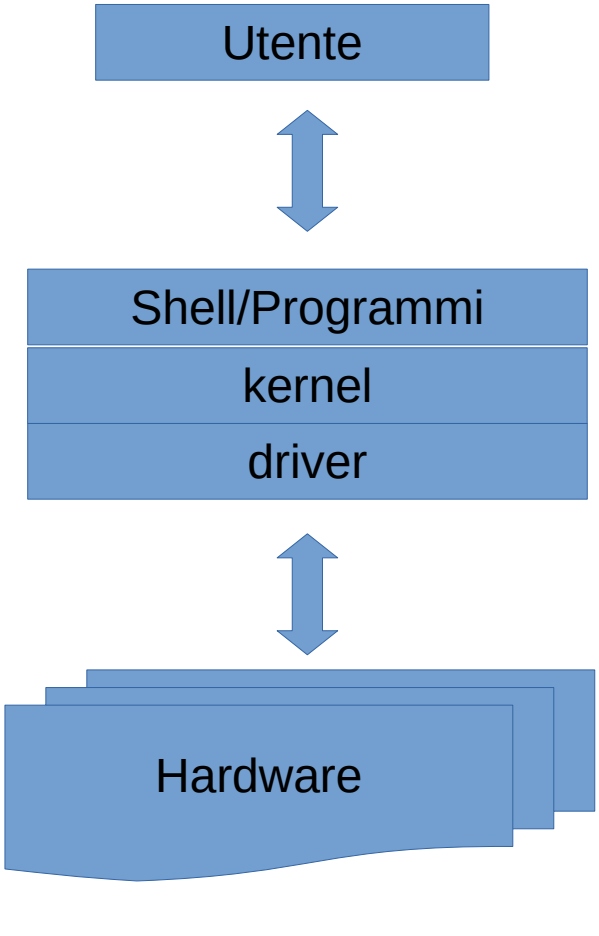
In questa lezione

- Sistemi Operativi
- Kernel
- Shell
- Programmi e Processi
- Time Sharing e Scheduler
- Gestione della memoria
- File System

Il Sistema Operativo

- Abbiamo visto come il sistema operativo sia il software principale in un calcolatore, responsabile di fornire adeguato supporto ai programmi e di gestire le risorse HW idealmente “sottostanti”
- A sua volta si può rappresentare il sistema operativo con un modello a strati che ha nel suo strato più basso la componentistica del calcolatore e nel suo strato più alto l’utente

Il Sistema Operativo



Kernel

Kernel è il nucleo del sistema operativo:

- è responsabile della gestione delle risorse
- si occupa di fornire ai processi in esecuzione un accesso SICURO e CONTROLLATO all'hardware

Esistono diverse tipologie di kernel, basate sostanzialmente su diversi concetti di design, ciascuno con suoi pregi e difetti.

Tipi di kernel

- Kernel monolitici, che implementano direttamente una completa astrazione dell'hardware sottostante.
- Microkernel, che forniscono un insieme ristretto e semplice di astrazione dell'hardware e usano software (chiamati device driver o server) per fornire maggiori funzionalità
- Kernel ibridi (o microkernel modificati), che si differenziano dai microkernel puri per l'implementazione di alcune funzioni aggiuntive al fine di incrementare le prestazioni

Shell

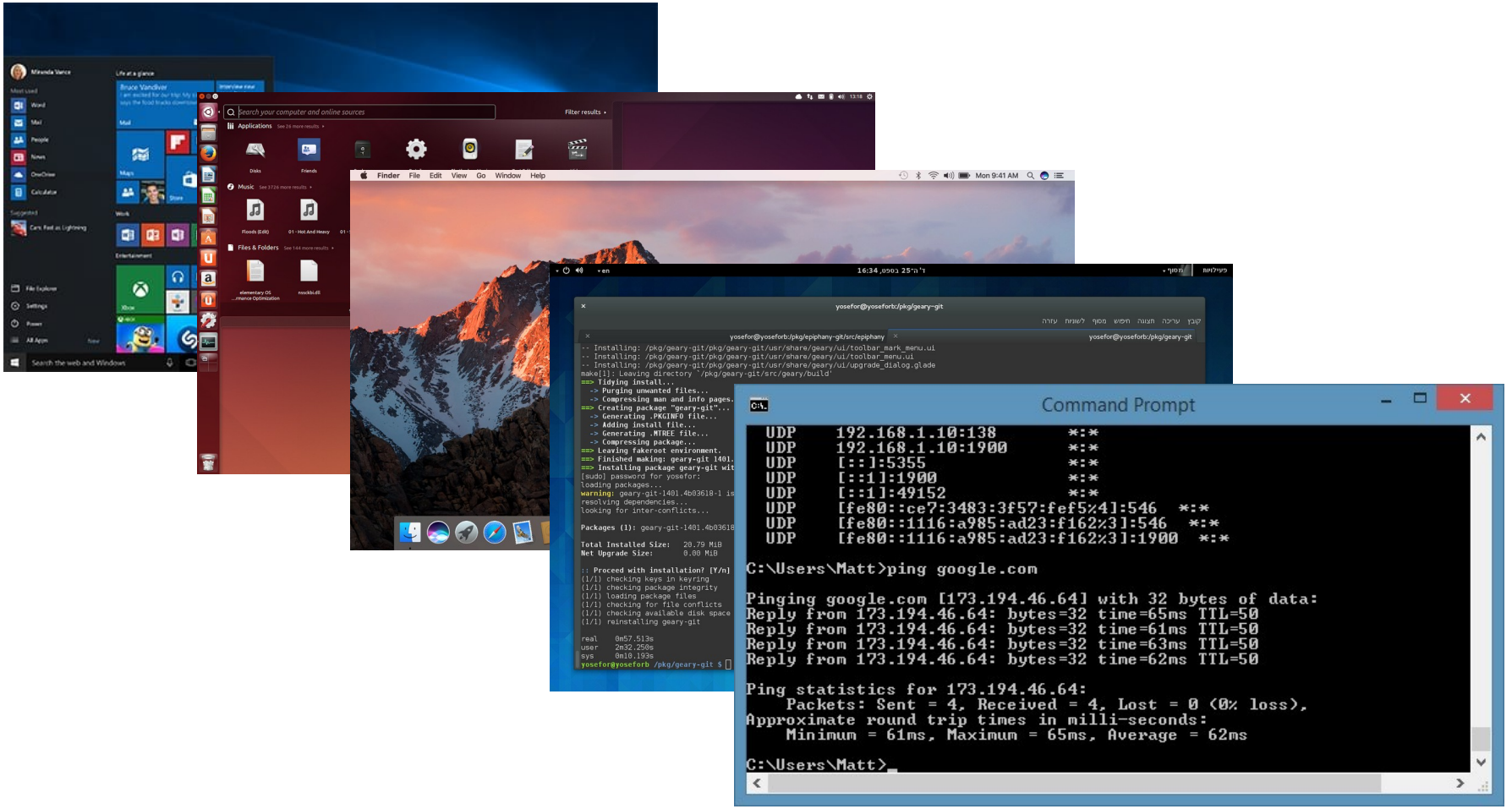
Shell è l'interprete dei comandi, ovvero l'interfaccia con cui interagisce l'utente.

E' così chiamata (shell=involucro) in quanto è la parte del Sistema Operativo **visibile** all'utente.

Esistono diversi tipi di shell, la distinzione più comune è:

- Shell grafiche (e.g. l'interfaccia utente di Windows)
- Shell testuali (e.g. il prompt dei comandi cmd.exe)

Shell


Command Prompt
UDP 192.168.1.10:138 *:*
UDP 192.168.1.10:1900 *:*
UDP [::1]:5355 *:*
UDP [::1]:1900 *:*
UDP [::1]:49152 *:*
UDP [fe80::ce7:3483:3f57:fef5%4]:546 *:*
UDP [fe80::1116:a985:ad23:f162%3]:546 *:*
UDP [fe80::1116:a985:ad23:f162%3]:1900 *:*
C:\Users\Matt>ping google.com
Pinging google.com [173.194.46.64] with 32 bytes of data:
Reply from 173.194.46.64: bytes=32 time=65ms TTL=50
Reply from 173.194.46.64: bytes=32 time=61ms TTL=50
Reply from 173.194.46.64: bytes=32 time=63ms TTL=50
Reply from 173.194.46.64: bytes=32 time=62ms TTL=50
Ping statistics for 173.194.46.64:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 61ms, Maximum = 65ms, Average = 62ms
C:\Users\Matt>

Programma

Un Programma è l'insieme delle istruzioni e dei dati definiti dal programmatore. È un'entità statica tipicamente memorizzata su un supporto di archiviazione (disco). Si distinguono:

- *Programmi interattivi*: programmi che durante la loro esecuzione dialogano con l'utente
- *Programmi batch*: programmi che non hanno interazione con l'utente, eseguiti in background

Entrambi sono eseguiti tramite l'ausilio del S.O.

Processi

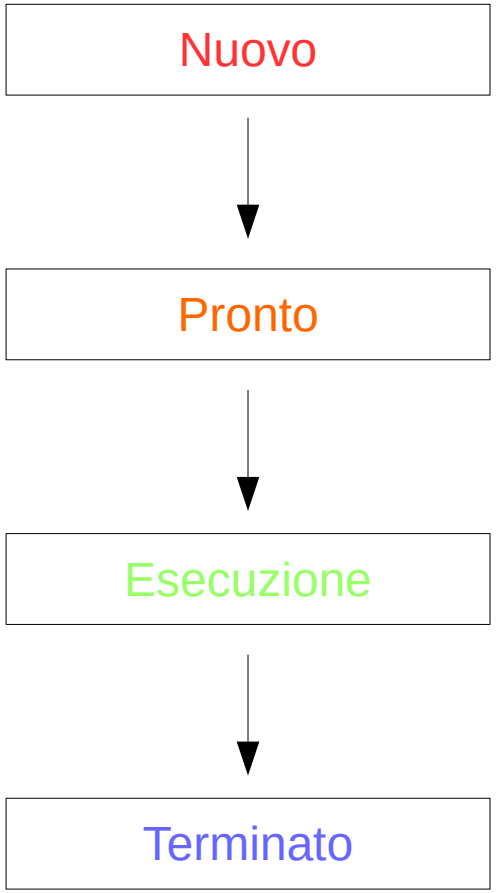
Abbiamo visto che con il termine processo si indica tipicamente un programma in esecuzione.

Ovvero è lo stato del programma dopo essere cambiato da **entità statica** presente in una memoria di massa a **entità in continuo mutamento** dopo essere stata copiata nella memoria principale ed eseguita

Una descrizione più formale di processo è pertanto:

- Una copia di un programma in esecuzione sul calcolatore più i dati necessari alla sua esecuzione

Processi



Quando un processo viene creato, gli viene assegnato lo stato Nuovo e rimane in questo stato fino a quando non gli vengono assegnate tutte le risorse di cui ha bisogno per essere eseguito

Quando ad un processo vengono assegnate tutte le risorse tranne il processore, questo passa allo stato Pronto

Quando al processo viene assegnato anche il processore, allora il processo passa in Esecuzione

Quando un processo termina tutte le operazioni che deve compiere allora passa nello stato Terminato e le risorse che gli erano state dedicate vengono liberate

Esempio

Ad esempio, l’utente può richiede al sistema operativo, interagendo con la shell, di eseguire un programma (es. facendo un “doppio click” sull’icona corrispondente).

A questo punto, il sistema operativo, svolge le seguenti attività:

1. individua il codice eseguibile del programma (residente sul disco)
2. alloca al programma le risorse necessarie per la sua esecuzione (ne consente l’utilizzo)
3. carica il codice eseguibile del programma nella memoria centrale
4. avvia l’esecuzione del programma

Sistemi monoprogrammati e multiprogrammati

Sistemi monoprogrammati

In memoria può risiedere al più un programma (processo) oltre al S.O.

Esempio: MS-DOS

Vantaggio: Ridotto uso della CPU

Sistemi multiprogrammati

In memoria possono risiedere più programmi contemporaneamente oltre al S.O.

Esempio: Unix

Vantaggi: Uso più efficiente della CPU

- I moderni S.O. sono tutti multiprogrammati: è importante sottolineare che l'esecuzione contemporanea in un sistema dotato di una sola CPU **è sempre virtuale**

Multitasking

In informatica, un sistema operativo con supporto per il multitasking permette di eseguire più programmi contemporaneamente

La percezione del multitasking

Per un attimo accantoniamo i processori della odierna generazione tecnologica e torniamo a qualche anno fa (Pentium 4):

La maggior parte dei processori desktop era formata da una sola unità di elaborazione, nonostante questo la percezione dell'utente nell'utilizzo quotidiano non è cambiata poi molto in questi anni...

La percezione del Multitasking

Domanda:

- Come fa un computer con una sola unità di elaborazione a far percepire all'utente che sta svolgendo diverse attività **contemporaneamente**?

Time Sharing

I sistemi a partizione di tempo (time sharing) simulano un quasi-parallelismo nell'accesso alle risorse.

Supponiamo di avere diversi programmi in memoria, ognuno con un diverso tempo di esecuzione: per evitare che un programma monopolizzi il processore, il tempo viene suddiviso in unità elementari, dette **quanti**, da assegnare ai vari processi secondo una possibile politica.

Come potete immaginare la dimensione del quanto di tempo incide fortemente sulle prestazioni del sistema.

La percezione del multitasking 2

In un Sistema Operativo sarà quindi presente una importantissima componente software avente il compito di gestire le politiche di assegnamento delle risorse ai processi richiedenti.

➔ Questo componente software è detto **SCHEDULER**

Scheduler (S.O.)

Parte del S.O. che programma le attività assegnando la CPU ai programmi in esecuzione.

Si basa su una **ripartizione equa** del tempo di occupazione della CPU.

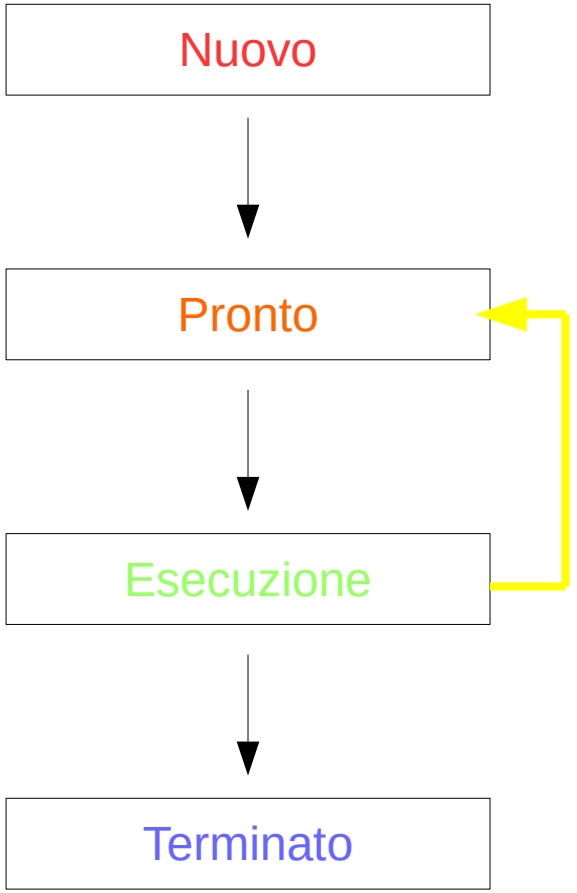
- Assegna una priorità maggiore ai programmi interattivi rispetto a quelli in background.
- Regola la priorità dei programmi in esecuzione per evitare l'eterna attesa.

Processi: Multitasking

Come si integra allora l'analisi vista prima degli stati di un processo in un sistema multitasking?

- Se un processo non ha finito tutte le sue operazioni, al termine del periodo di esecuzione assegnato torna in stato di "Pronto" per continuare l'esecuzione in un secondo momento

Processi: Multitasking



- Se il processo non termina tutte le operazioni allora ritorna allo stato Pronto, le risorse restano allocate al processo **ma il processore passa ad eseguire una latro processo**
- Il processo poi torna di nuovo in Esecuzione

Gestione della memoria

E’ la componente del sistema operativo che si occupa di gestire ed assegnare la memoria ai processi che ne fanno richiesta.

La gestione della memoria è necessaria per tenere traccia di quanta memoria è impegnata e di quanta invece è disponibile per soddisfare nuove richieste: in mancanza di un sistema di gestione, si avrebbe prima o poi il caso di processi che ne sovrascrivono altri, con ovvi inconvenienti.

Un altro buon motivo per registrare la memoria usata dai vari processi è il fatto che in caso di errori gravi i processi possono andare in crash e non essere più in grado di comunicare al sistema che la memoria che occupano può essere liberata: in questo caso è compito del gestore di memoria, dopo la terminazione anomala del processo, marcare come libere le zone di memoria possedute dal processo andato in crash, rendendole disponibili per nuove allocazioni.

Gestione della memoria

- Il gestore della memoria consente l'allocazione dinamica della memoria centrale (ram) ai programmi in esecuzione
- A ciascun programma viene allocata un'area di memoria virtuale, sufficiente per la sua esecuzione
- Il gestore della memoria gestisce la corrispondenza tra le memorie virtuali e l'unica memoria reale
- La dimensione della memoria virtuale può essere maggiore di quella reale: i dati possono essere parcheggiati temporaneamente nella memoria secondaria (disco fisso)

Gestione della memoria: swap

Nei S.O. multiprogrammati possiamo avere molti processi in memoria centrale.

Problema: La memoria centrale e' tipicamente di piccole dimensioni (Qualche Gb), come possiamo eseguire molti processi?

Si rende necessaria l'implementazione di un sistema che scarichi in memoria di massa i processi poco utilizzati e che inoltre permetta di ridurre la necessità di spazio mantenendo in memoria solo parte delle istruzioni e dei dati del processo.

Ridurre la necessità di spazio condividendo il codice fra diversi processi che corrispondono allo stesso programma (segmentazione della memoria).

Gestione della memoria: swap

Il gestore della memoria permette quindi di trasferire il contenuto di un'area di memoria centrale in un'area di memoria di massa chiamata memoria di swap.

Si trasferiscono nell'area di swap i processi in attesa e alcuni dei processi in stato di pronto al fine di liberare memoria centrale.

Per lo scarico nell'area di swap si segue il **Principio di Località**:

- se la CPU sta eseguendo una data istruzione presente in memoria, vuol dire che con molta probabilità le prossime istruzioni da eseguire saranno ubicate nelle vicinanze di quella in corso

Gestione delle memoria virtuale

Il gestore della memoria si occupa anche di mappare (indirizzare) la memoria virtuale offerta ai programmi:

- Sulla memoria fisica
- Sui dischi rigidi del sistema (swap)

Senza che i programmi stessi o gli utenti debbano preoccuparsi di nulla.

File System

In pratica, il File System, gestisce come i file sono memorizzati e organizzati su un dispositivo di archiviazione.

Più formalmente, un File System è l'insieme dei tipi di dato astratti necessari per

- Memorizzazione
- Organizzazione gerarchica
- Manipolazione
- Navigazione
- Accesso
- Lettura

dei dati.

File System

Riassumendo, il File System consente di gestire I file sulla memoria di massa:

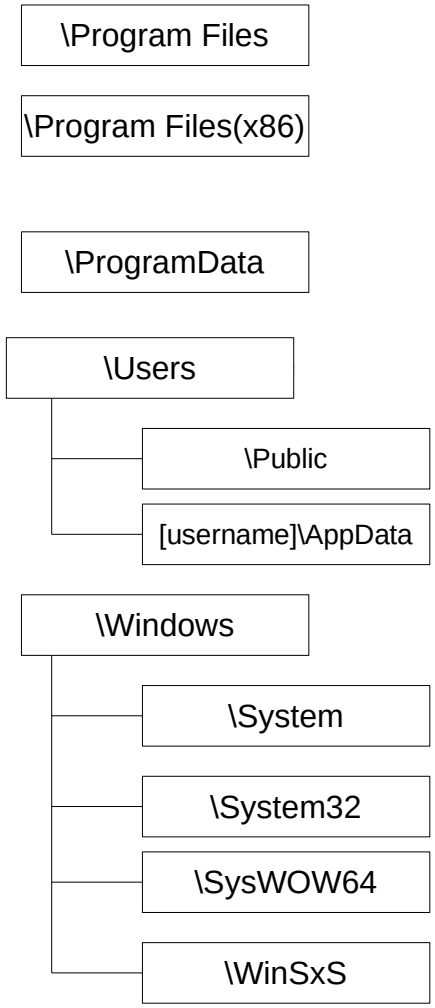
- Creare un file
- Dargli un nome
- Collocarlo in un opportuno spazio del supporto
- Accedervi in lettura e scrittura

indipendentemente dalle caratteristiche fisiche della memoria di massa.

I file vengono inclusi all'interno di directory ed hanno una

tipica organizzazione gerarchia detta "ad albero".

Organizzazione F.S. Windows



\Program Files: cartelle dei programmi installati sul sistema: 16, 32 e 64 bit

\ProgramData: dati e configurazioni delle applicazioni a prescindere da utenti e contesto (e.g. definizioni dei virus di Windows Defender)

\Users: cartelle personali degli utenti

- \Public: cartella dedicata alla condivisione dei file, l'accesso è consentito a tutti gli utenti sul sistema ed è condivisa in rete
- [username]\AppData]: dati e configurazioni delle applicazioni per utente

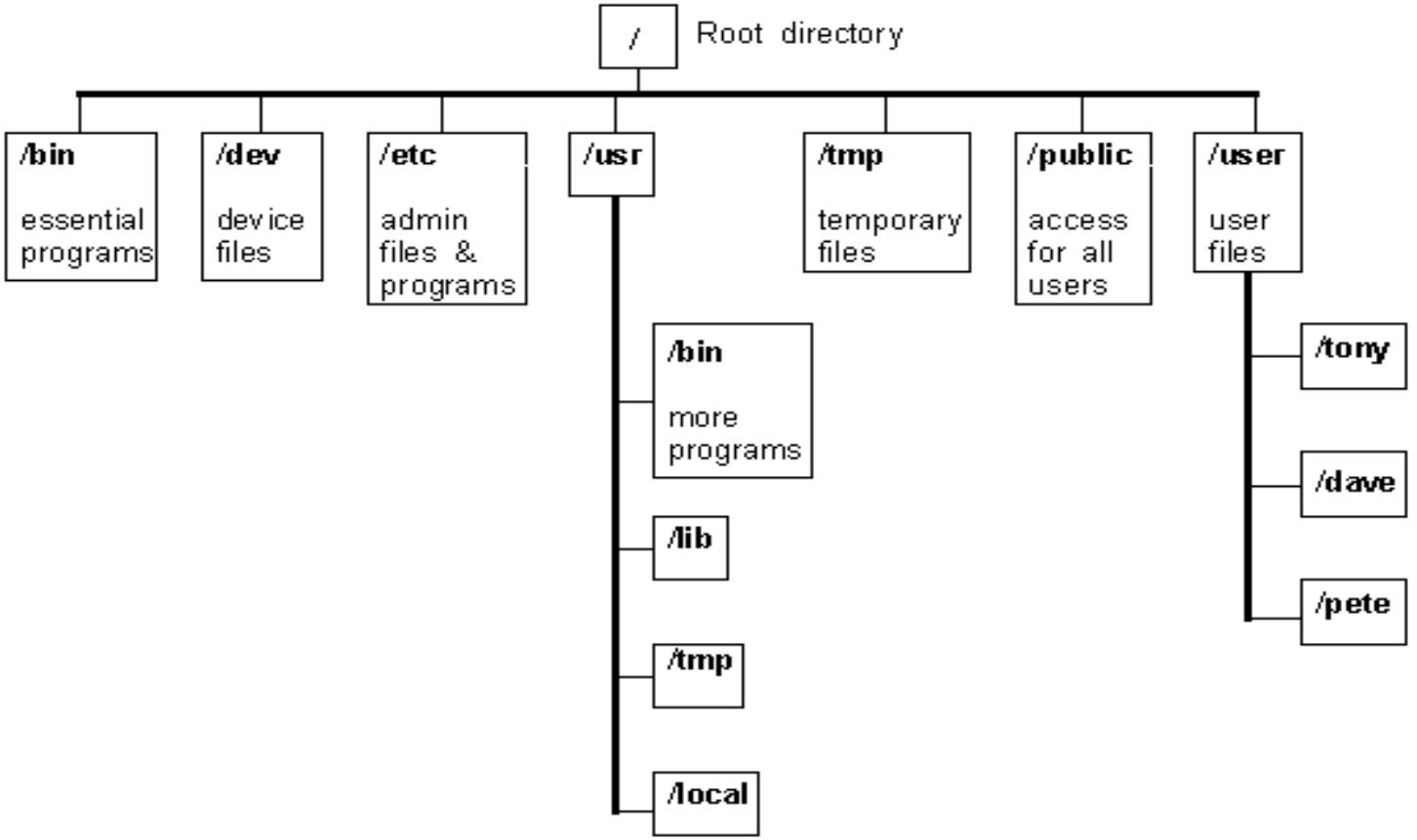
\Windows: cartella di installazione del sistema operativo: Windows è installato in questa cartella

Cartelle che contengono le librerie di sistema (dynamic-link library - DLL)

- System → 16-bit DLL
- System32 → 32/64 bit DLL (a seconda del sistema installato)
- SysWOW64 → DLL a 32 bit nei sistemi a 64 bit

Cartella delle componenti di Windows: sistema, aggiornamenti, service pack

Filesystem Hierarchy Standard (Unix/Linux)



Librerie

Una libreria, in Informatica, è un insieme di funzioni o strutture dati predefinite e predisposte per essere collegate ad un programma software attraverso opportuno collegamento.

Lo scopo delle librerie software è fornire una collezione di entità di base pronte per l'uso ovvero riuso di codice, evitando al programmatore di dover riscrivere ogni volta le stesse funzioni o strutture dati e facilitando così le operazioni di sviluppo e manutenzione.

API

Con Application Programming Interface si indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare set di strumenti specifici per l'espletamento di un determinato compito all'interno di un certo programma.

Spesso con tale termine si intendono le librerie software disponibili in un certo linguaggio di programmazione.

Chiamate di sistema

Una chiamata di sistema (system call) è il meccanismo usato da un programma a livello utente per richiedere un servizio a livello kernel del S.O.

Vi sono fondamentalmente due tipi di chiamate di sistema:

chiamate al kernel e **chiamate alle librerie utente**

Le categorie principali di system call sono:

- Controllo dei processori e dei job (end, abort, load, execute ecc).
- Manipolazione dei file e dei dispositivi (create, delete file, open, close ecc).
- Gestione delle informazioni.
- Comunicazione.

Input/Output

Per comunicare con l'utente i processi possono "sfruttare" 3 canali principali:

- standard input – leggere quello che l'utente scrive
- standard output – scrivere il risultato sullo schermo
- standard error – dove indirizzare i messaggi di errore

Un programma può anche accettare file come standard input e standard output.

Grazie per l’attenzione

Riferimenti

Il corso di programmazione per il primo anno della Laurea Triennale in Matematica nasce con l'intento di unire ai principi di programmazione una conoscenza basilare di uno degli strumenti software più diffusi nell'ambito matematico: Matlab.

La prima parte del corso pertanto è una rielaborazione del programma di Programmazione per la Laurea Triennale in Informatica 15/16 (in particolare) e 16/17.

Parte del materiale originale da cui ho ricavato il percorso didattico, alcuni approfondimenti ed integrazioni possono essere trovati in:

- Lezioni di Programmazione 15/16 CdS Informatica - Giacomo Piva
- Lezioni di Programmazione 16/17 CdS Informatica - Marco Alberti