Struttura dei Sistemi Operativi

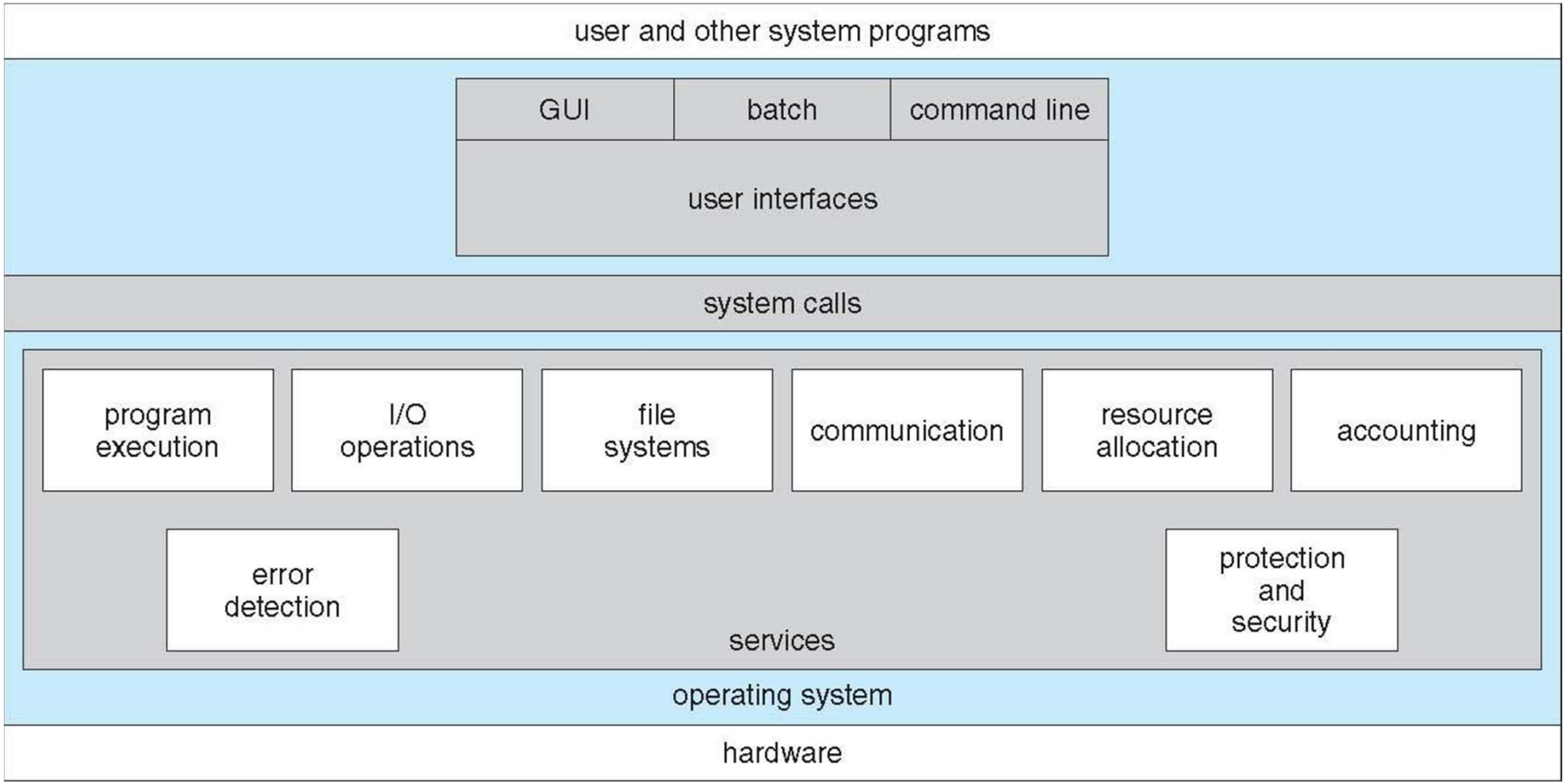
Capitolo 2 -- Silberschatz

Struttura di un sistema operativo

- Servizi di un sistema operativo
- Interfaccia Utente
- Chiamate di sistema
- Tipi di chiamate
- Programma di sistema

Servizi per l’utente generico

Servizi per l'utente generico



Servizi per l'utente generico

- **Esecuzione dei programmi** – Il SO carica in memoria ed esegue i programmi e ne termina l'esecuzione in modo normale o anormale, indicandone l'errore
- **Operazioni di I/O** - Un programma in esecuzione può richiedere I/O su file o dispositivi di I/O
- **Gestione del file-system** - I programmi necessitano di leggere e scrivere file e directory, crearne e cancellarne, accedere alle informazioni di gestione del file, gestire i permessi

Servizi per l'utente generico

- **Comunicazioni** – I processi possono scambiarsi informazioni, sullo stesso computer o tra computer su una rete
- **Rilevazione dell'errore** – il SO deve essere costantemente informato sugli errori
  - CPU, hardware della memoria, I/O device, programmi utente
  - Per ogni tipo di errore, il SO dovrebbe prendere le azioni appropriate per assicurare computazioni corrette e consistenti
  - Strumenti di supporto per il debug incrementano le possibilità dell'utente e dei programmatori di usare efficientemente il SO

Servizi impliciti

- **Allocazione risorse –** Quando più utenti o processi vengono eseguiti concorrentemente, le risorse debbono essere allocate ad ognuno di essi
  - Risorse : cicli CPU, memoria, file, dispositivi I/O
- **Contabilizzazione –** Tenere traccia di quali utenti (e quanto) usano certe risorse

Servizi impliciti

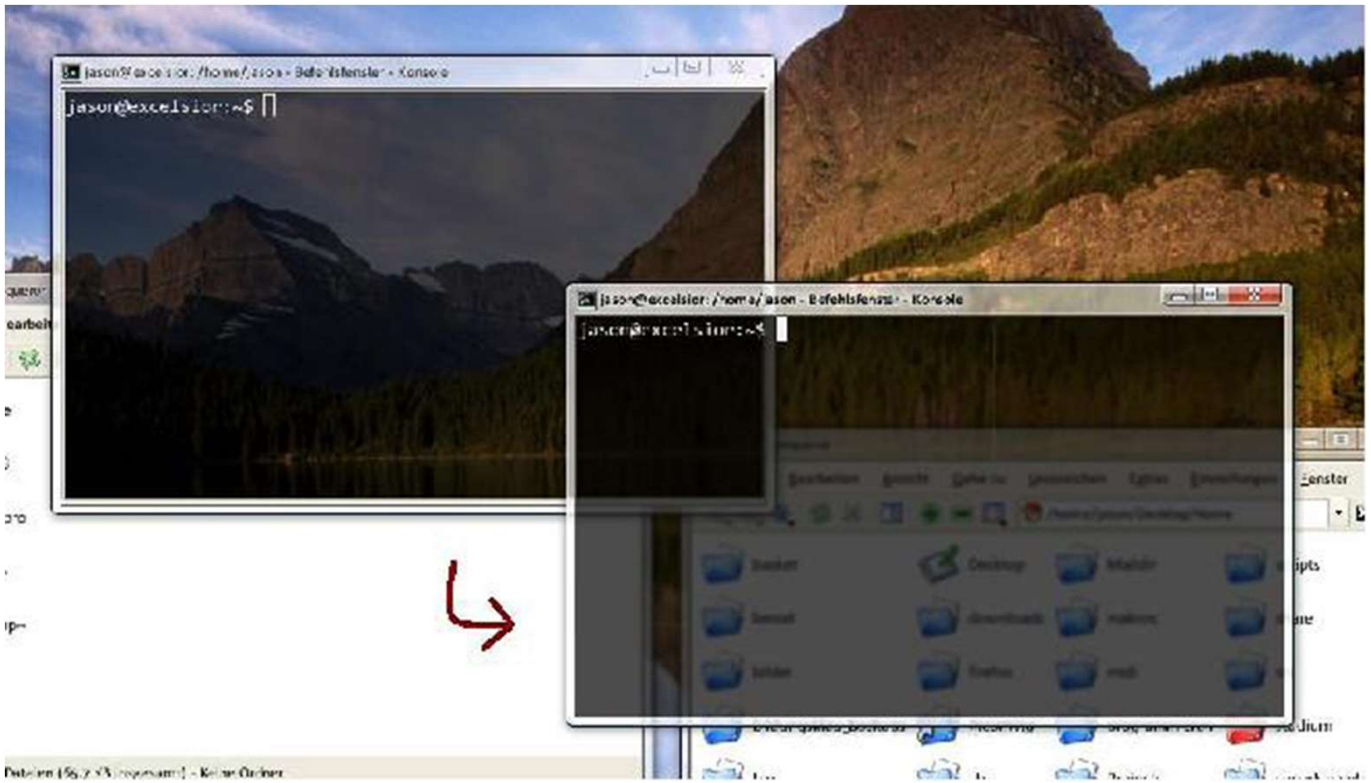
- **Protezione e sicurezza –** I proprietari di dati memorizzati in un sistema multiutente o connesso alla rete desiderano controllare l'uso dei propri dati ed esser sicuri che i processi non interferiscano
  - **Protezione:** tutti gli accessi alle risorse del sistema sono controllati
  - **Sicurezza:** autenticazione degli utenti contro attacchi esterni
  - "una catena è tanto forte quanto il suo anello più debole".

Interfaccia utente CLI

- Le interfacce a linea di comando CLI permettono l'invio diretto di comandi
  - Caratteristiche multiple e opzioni – **shell**
  - A volte i comandi sono built-in (implementati nel kernel), altre sono semplici nomi di eseguibili
  - Con il secondo approccio, l'aggiunta di nuove caratteristiche non richiede la modifica della shell

Unix: Bourne Shell, C Shell, Bourne-again Shell, Korn Shell

Interfaccia utente CLI per Unix

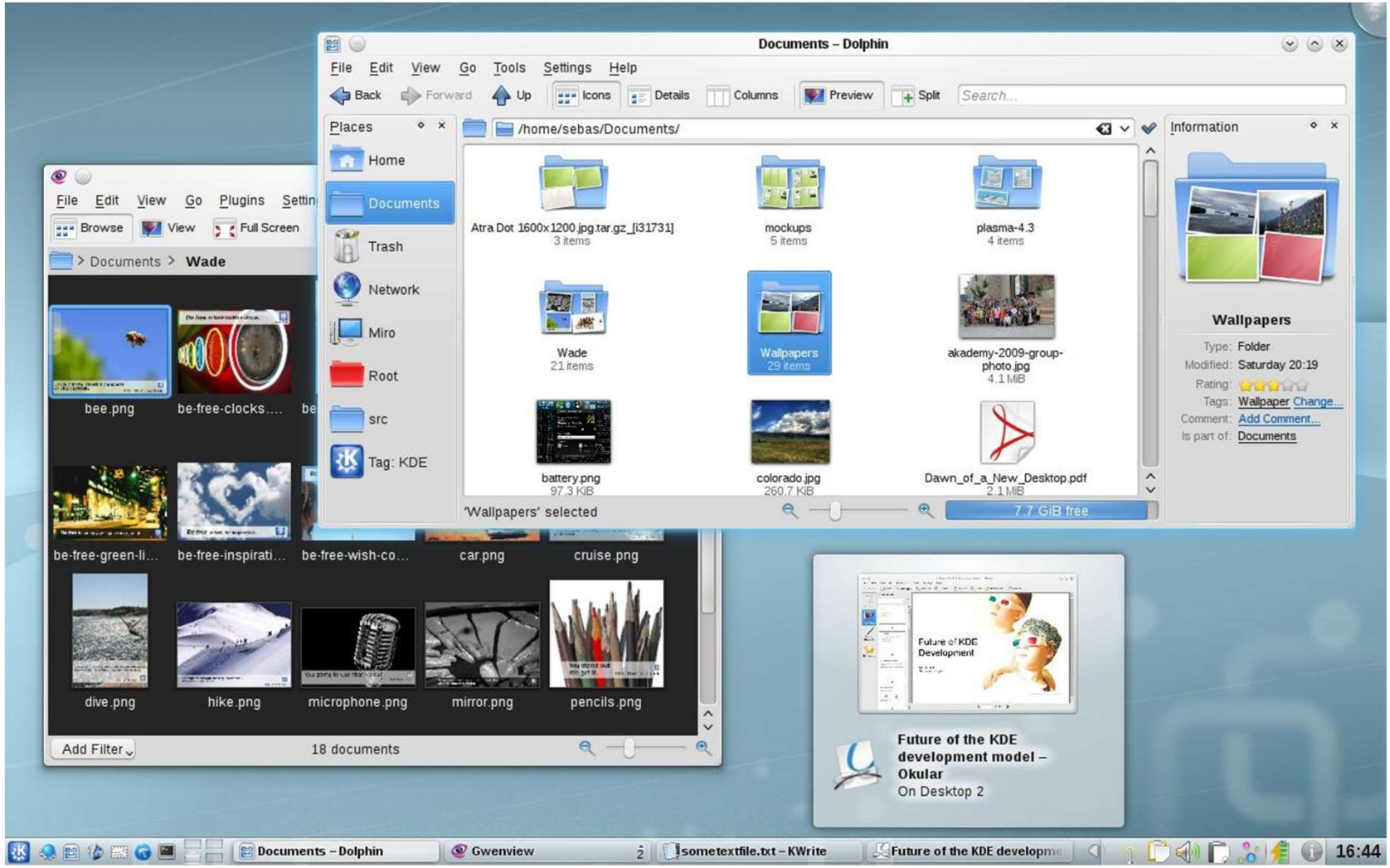


Bash shell (per Bourne Again SHell) è una shell testuale del progetto GNU

Interfaccia utente GUI

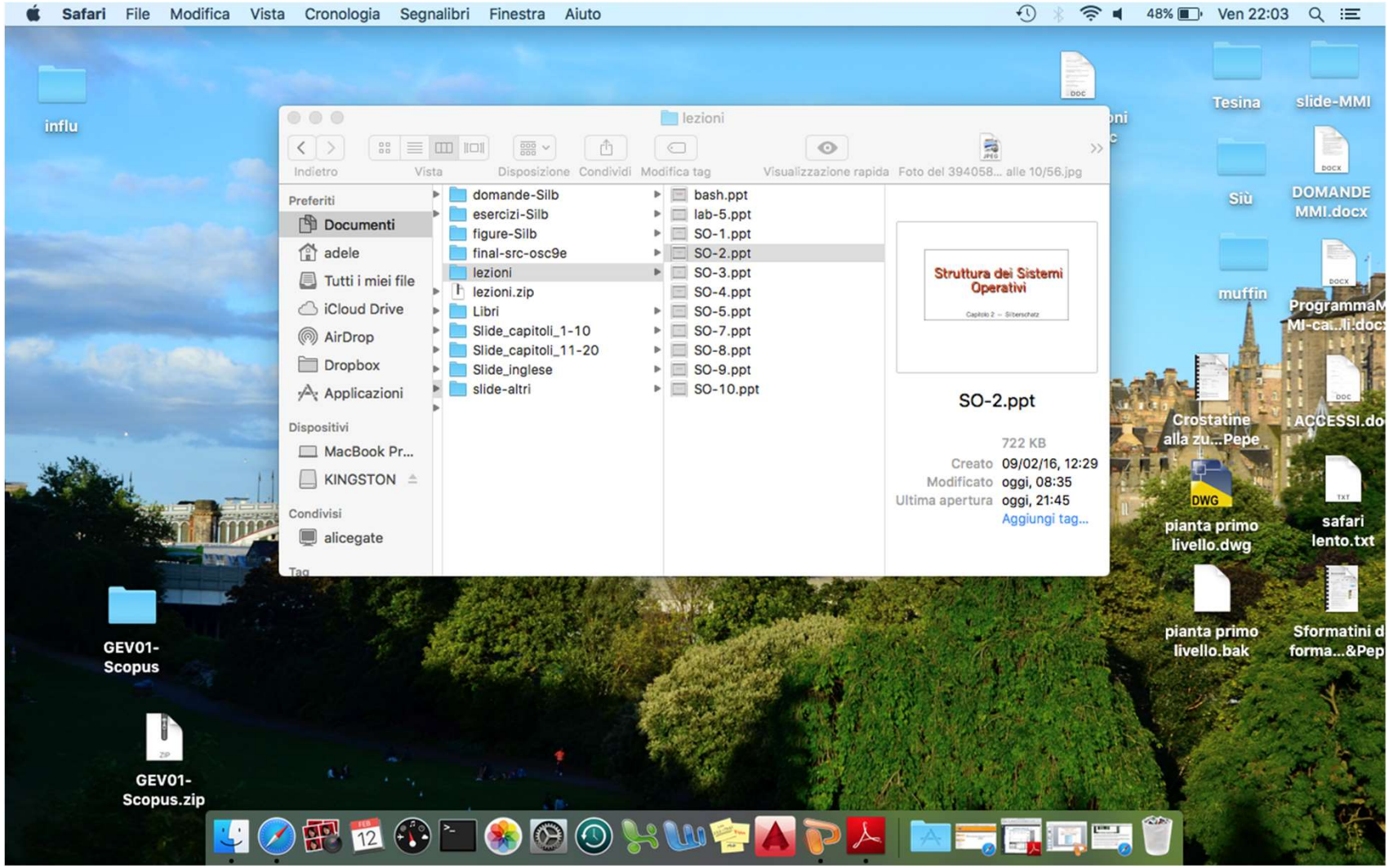
- Le interfacce grafiche forniscono **desktop** amichevoli
  - I comandi sono forniti tramite mouse, tastiera, monitor
  - **Icone** rappresentano file, programmi, azioni ...
  - La pressione dei tasti del mouse sugli oggetti dell'interfaccia causa varie azioni (recupero informazioni, opzioni, esecuzioni di funzioni, apertura di directory)
  - Inventate a Xerox PARC, comuni con Apple Mac

Interfaccia utente GUI



il desktop di GNU/Linux

Interfaccia utente GUI



GUI di Mac OS X

Interfaccia utente

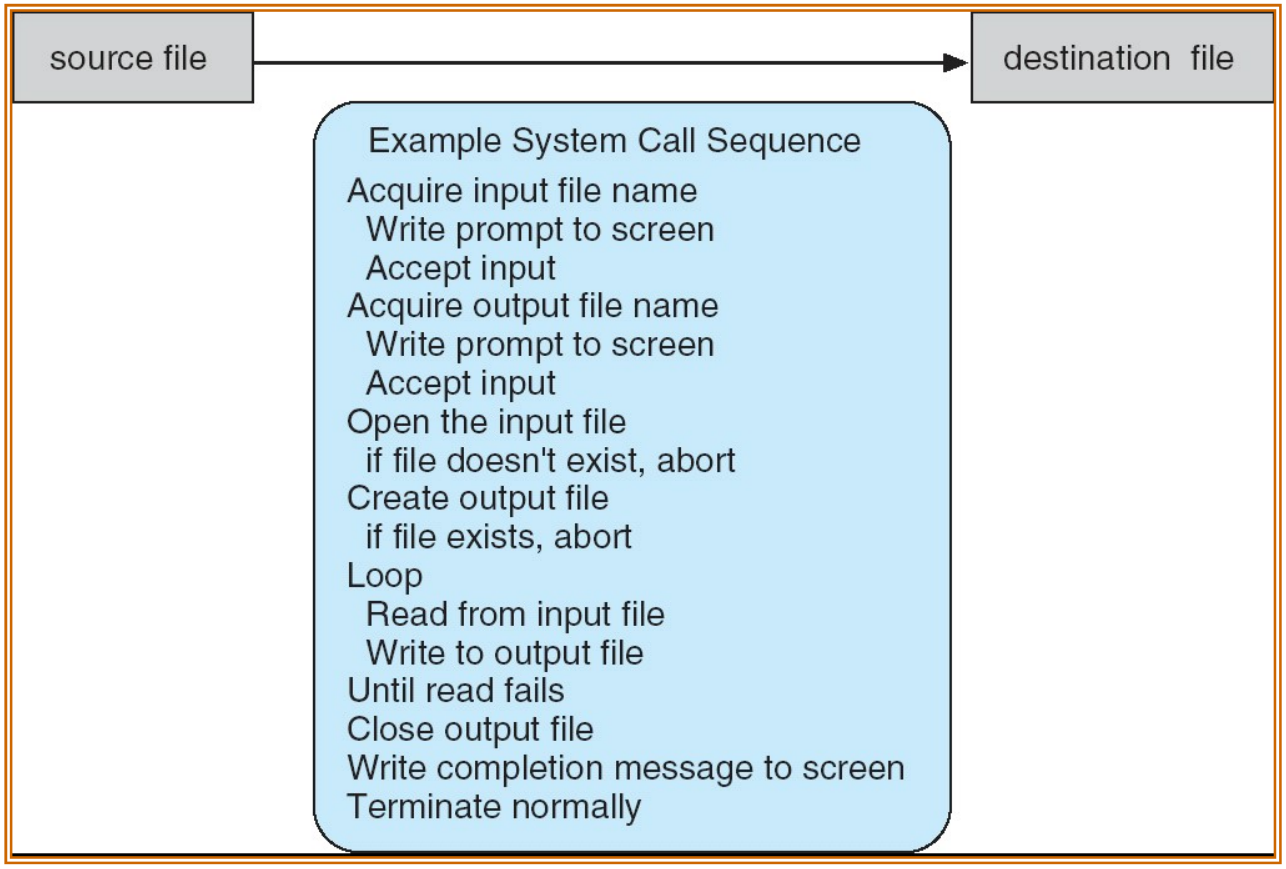
Molti sistemi oggi includono interfacce sia CLI che GUI

- Microsoft Windows principalmente basato su una interfaccia grafica, ma dotato anche di una shell di comandi DOS-based (cmd)
- Apple Mac OS X offre una GUI (Aqua), che poggia su un kernel UNIX, e mette a disposizione le shell UNIX
- Solaris offre una CLI e opzionalmente GUI (Java Desktop, KDE)
- Linux è modulare; si può scegliere tra GUI molto avanzate (KDE, GNOME, etc.) e la CLI

Servizi per il programmatore

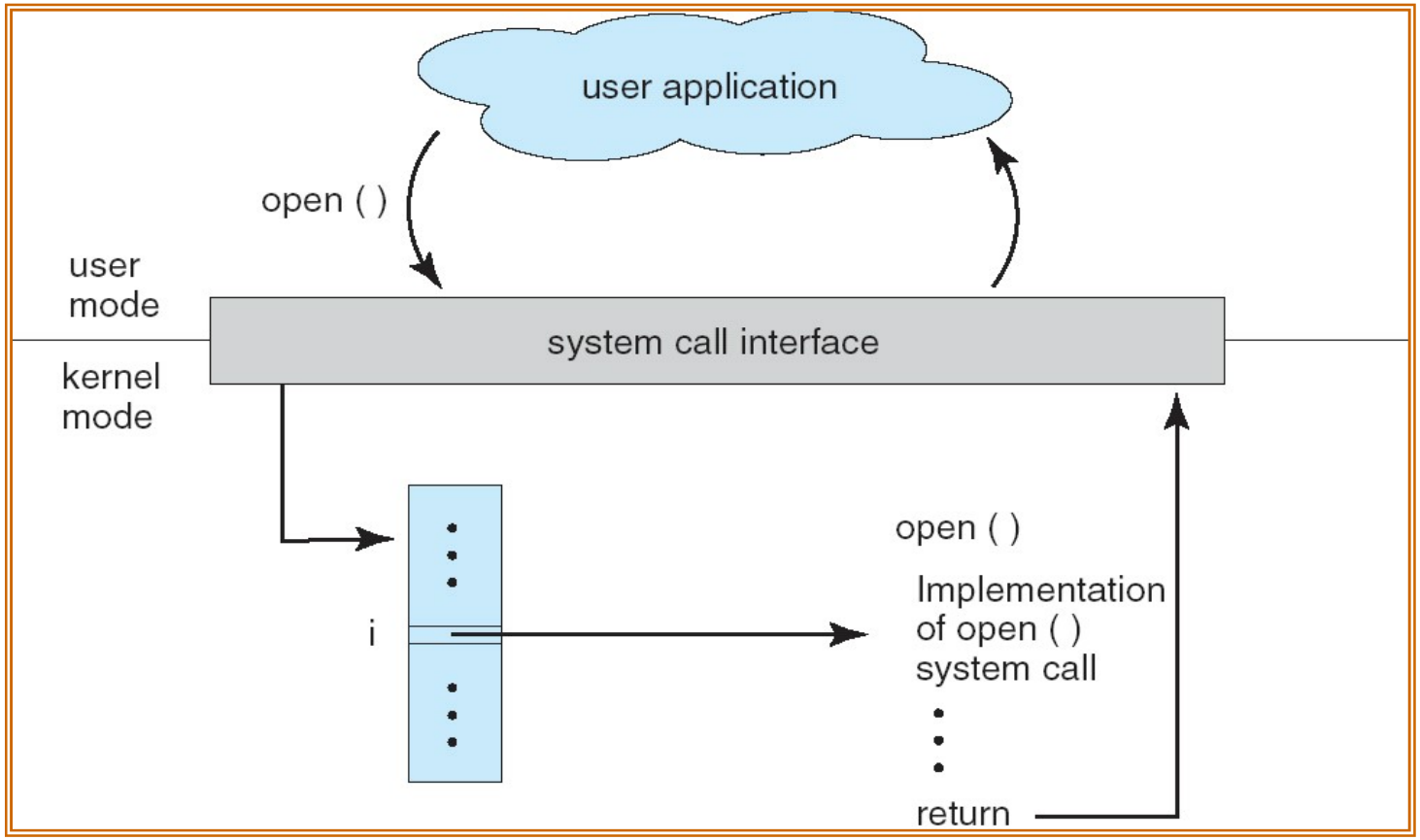
Chiamate di sistema
System Call

- Interfaccia per i programmatori ai servizi offerti dal SO
- Tipicamente scritte in un linguaggio ad alto livello (C or C++)



Gestione chiamate di sistema

- Tipicamente ad ogni chiamata di sistema è associato un numero
  - Il sistema mantiene una tabella indicizzata da questi numeri

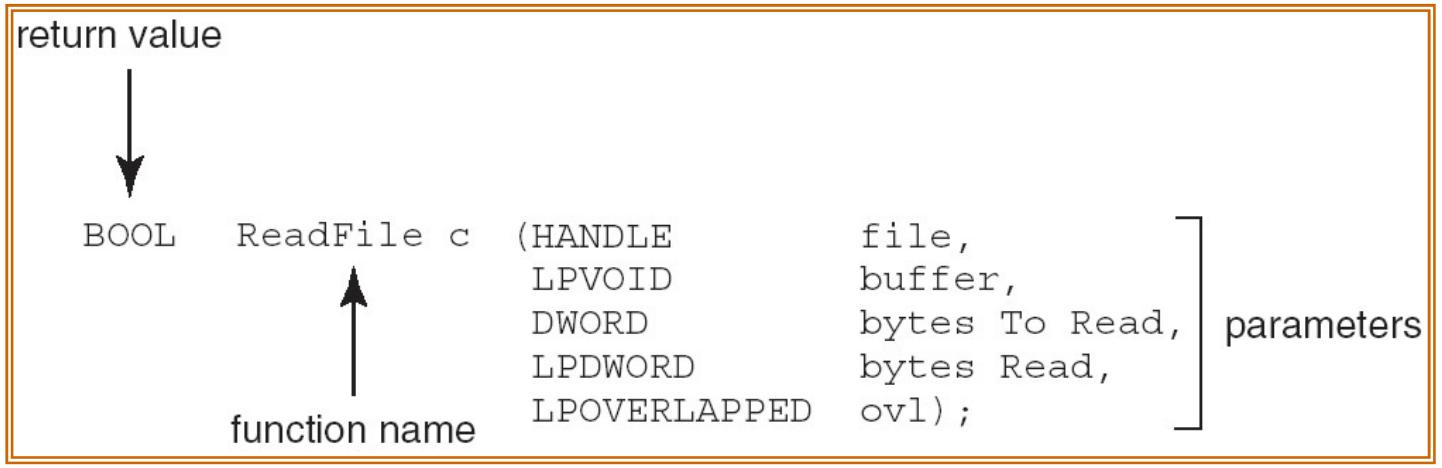


API e Chiamate di sistema

- I programmi accedono alle chiamate di sistema principalmente tramite **Application Programming Interface (API)** piuttosto che chiamate dirette
- Tre API comuni sono la **Win32 API** per Windows, la **POSIX API** per sistemi POSIX-based (che virtualmente includono tutte le versioni di UNIX, Linux, e Mac OS X), e la **Java API** per la Java virtual machine (JVM)
- Perchè usare API invece di system call direttamente? Portabilità dei programmi, dettagli delle system call reali ...

Esempio API

- Considera la funzione ReadFile() - Win32 API
- I parametri passati a ReadFile() sono
  - HANDLE file — il file da leggere
  - LPVOID buffer— un buffer in cui I dati vengono letti
  - DWORD bytesToRead— numero di byte da leggere nel buffer
  - LPDWORD bytesRead— numero di byte letti durante l'ultima read
  - LPOVERLAPPED ovl—indica se l' I/O prevede la sovrapposizione



API – System Call – Relazione con il SO

- L'interfaccia (API) alle chiamate di sistema invoca l'opportuna chiamata di sistema e restituisce lo stato e qualunque valore di ritorno della chiamata stessa
- Il chiamante deve semplicemente
  - seguire le specifiche dell'API e capire cosa il SO fa a seguito di una chiamata
- l'API nasconde al programmatore molti dettagli implementativi, gestiti dal sistema di supporto run-time (insieme di funzioni della libreria inclusa con il compilatore)

Categorie di system call

- Controllo dei processi
- Gestione dei file
- Gestione dei dispositivi
- Gestione delle informazioni di stato
- Comunicazioni

Categorie di system call

Processi

- end, abort
- load, execute
- create process, terminate process
- get process attribute, set process attributes
- wait for time
- wait event, signal event
- allocate memory, free memory

Dispositivi I/O

- request device, release device
- read, write, reposition
- get device attribute, set device attributes
- attach device, detach device

File

- create file, delete file
- open, close
- read, write, reposition
- get file attribute, set file attributes

Informazioni

- get time, get date, set time, set date
- get system data, set system data
- get process, file, or device attributes
- set process, file, or device attributes

Categorie di system call

Comunicazioni:

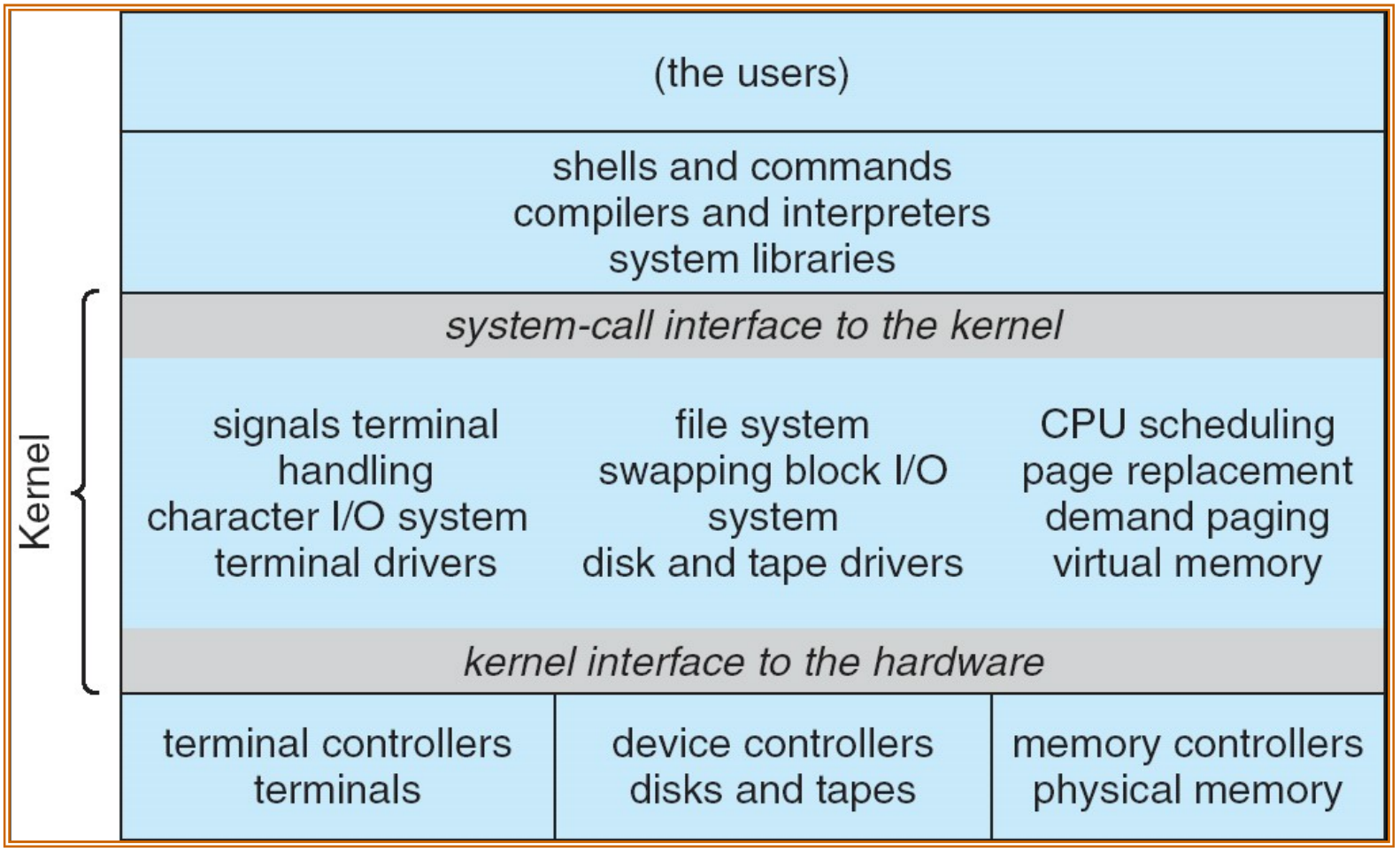
Scambio di massaggi

- gethostid – IP address
- getprocessid – PID
- open connection
- accept connection
- send/receive
- close connection

Memoria condivisa

- shared memory create
- shared memory attach

Struttura del sistema operativo UNIX

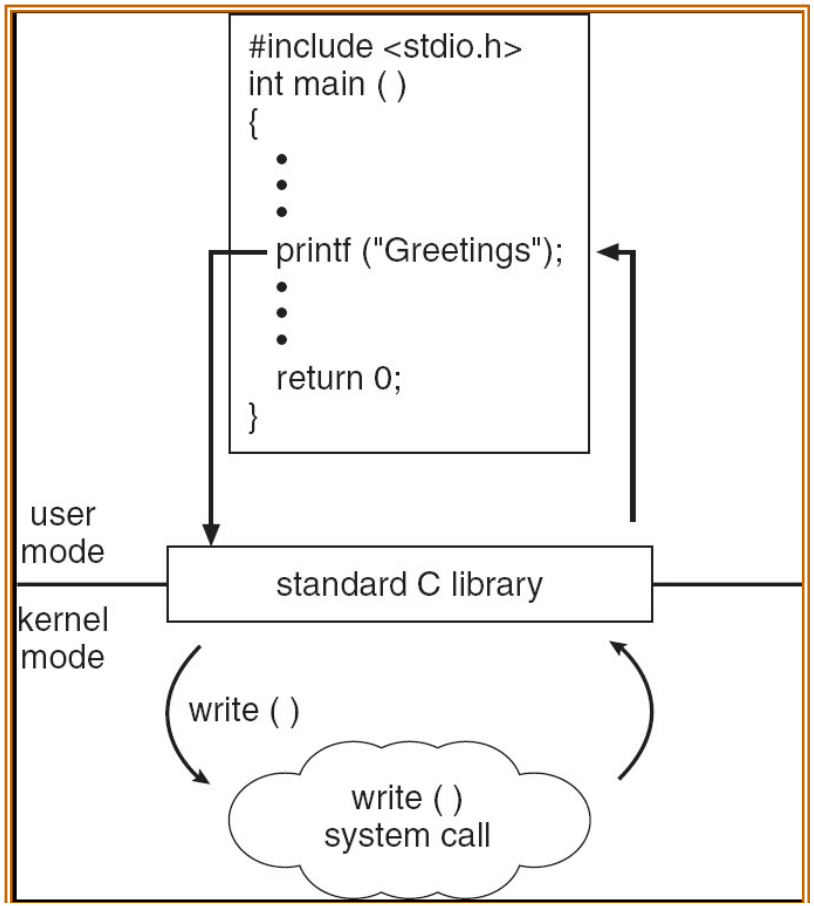


System Call & librerie in UNIX

- Una system call è un entry point del kernel per fornire servizi ai processi che li richiedono
  - comando shell man 2 fornisce la documentazione sulle system call (definite in C)
- system call → funzione omonima nella libreria standard (wrapper)
  - Es. ssize_t write (int *filedes*, void **buff*, size_t *nbytes*);
- l'utente chiama il wrapper (attraverso la sequenza standard di chiamate a funzioni di C), questo invoca il servizio del kernel
- Semplifichiamo: System Call = Funzioni C

Esempio della libreria C Standard

- Un programma C che invoca la funzione di libreria printf(), la quale ha bisogno di chiamare la system call write()



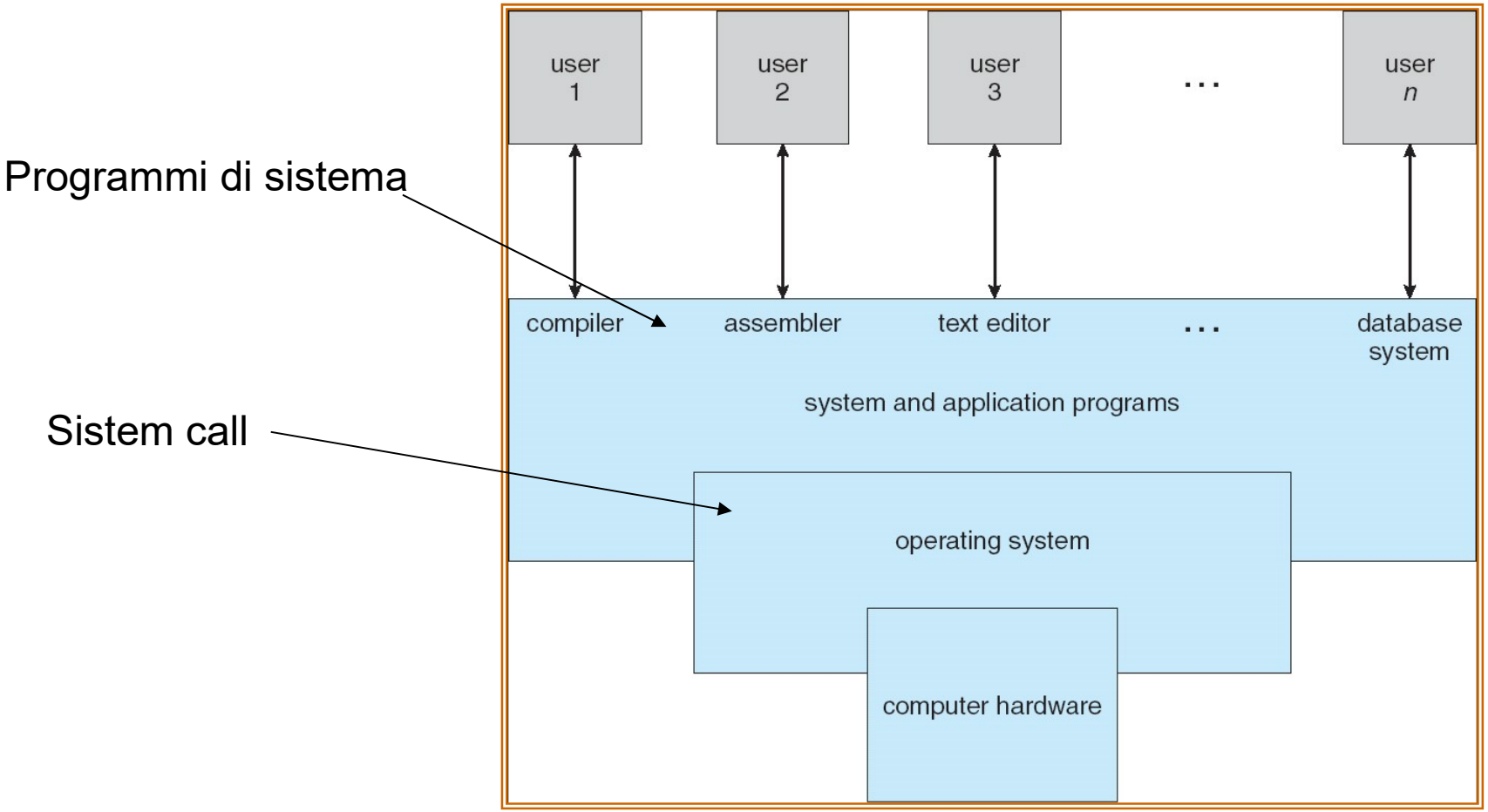
System Call & Librerie

- system call: interfaccia minima, mentre le funzioni di libreria forniscono più elaborate funzionalità
- libc: interfaccia normale

- system call sono definitive dal s.o., le funzioni di libreria no!

Programmi di sistema

- Alcuni sono semplici interfacce alle chiamate di sistema. Altri sono considerevolmente più complessi.



Programmi di sistema

- **Gestione dei File** - Create, delete, copy, rename, print, dump, list, e programmi per manipolare file e directory
- **Gestione informazioni di stato**
  - Alcuni chiedono al SO informazioni quali data, ora, quantità di memoria disponibile, spazio su disco, numero di utenti
  - Altri forniscono dettagli sulle prestazioni, informazioni sulle operazioni di log e il debug
  - Tipicamente, questi programmi formattano e stampano le informazioni al terminale o su altri dispositivi di output
  - Alcuni sistemi implementano un file registro, usato per memorizzare e recuperare informazioni di configurazione del SO

Programmi di sistema

- **Editing dei File**
  - Editor di testo per creare e modificare file
  - Comandi speciali per cercare contenuti di file o elaborare il testo
- **Supporto ai linguaggi di programmazione** - Compilatori, assemblatori, debugger e interpreti
- **Caricamento ed esecuzione dei programmi** – Caricatori assoluti, caricatori rilocabili, linker, e caricatori di overlay, sistemi di debug per linguaggi ad alto livello
- **Comunicazioni** – Forniscono un meccanismo per creare connessioni virtuali tra processi, utenti, e sistemi di calcolatori
  - Permettono agli utenti di inviare messaggi ad un altro schermo, sfogliare pagine web, inviare messaggi di posta elettronica, login remoto, trasferire file da una macchina ad un'altra