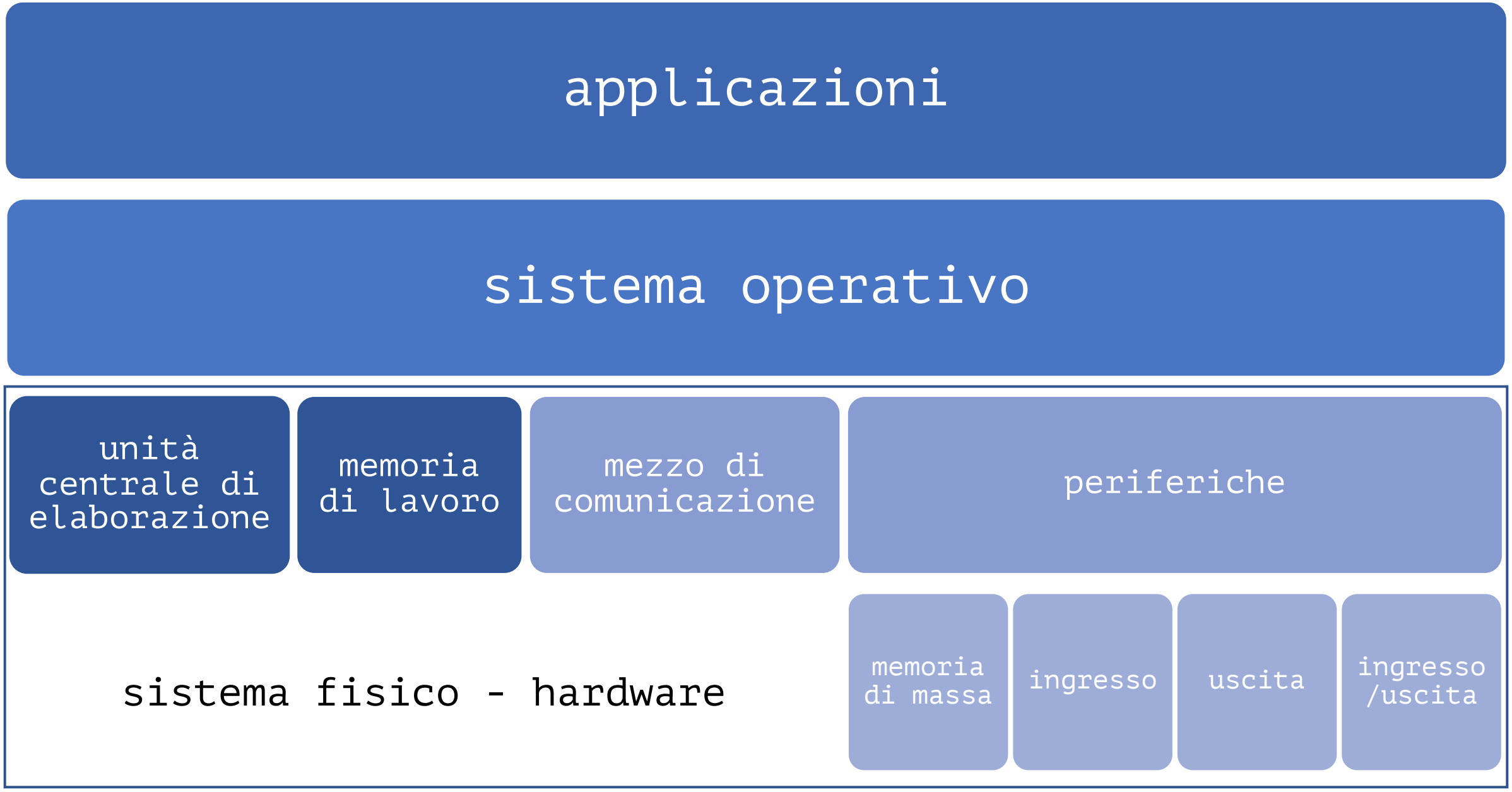
Architettura del sistema di calcolo

sistema operativo

visione astratta & semplificata

Cristiana Bolchini

sistema di calcolo



obiettivi

visione astratta e semplificata

capire come è organizzata un'architettura di un sistema di calcolo, con riferimento al modello di Von Neumann

- come funziona
- come è programmato
- come avviene l'esecuzione di un programma

introduzione

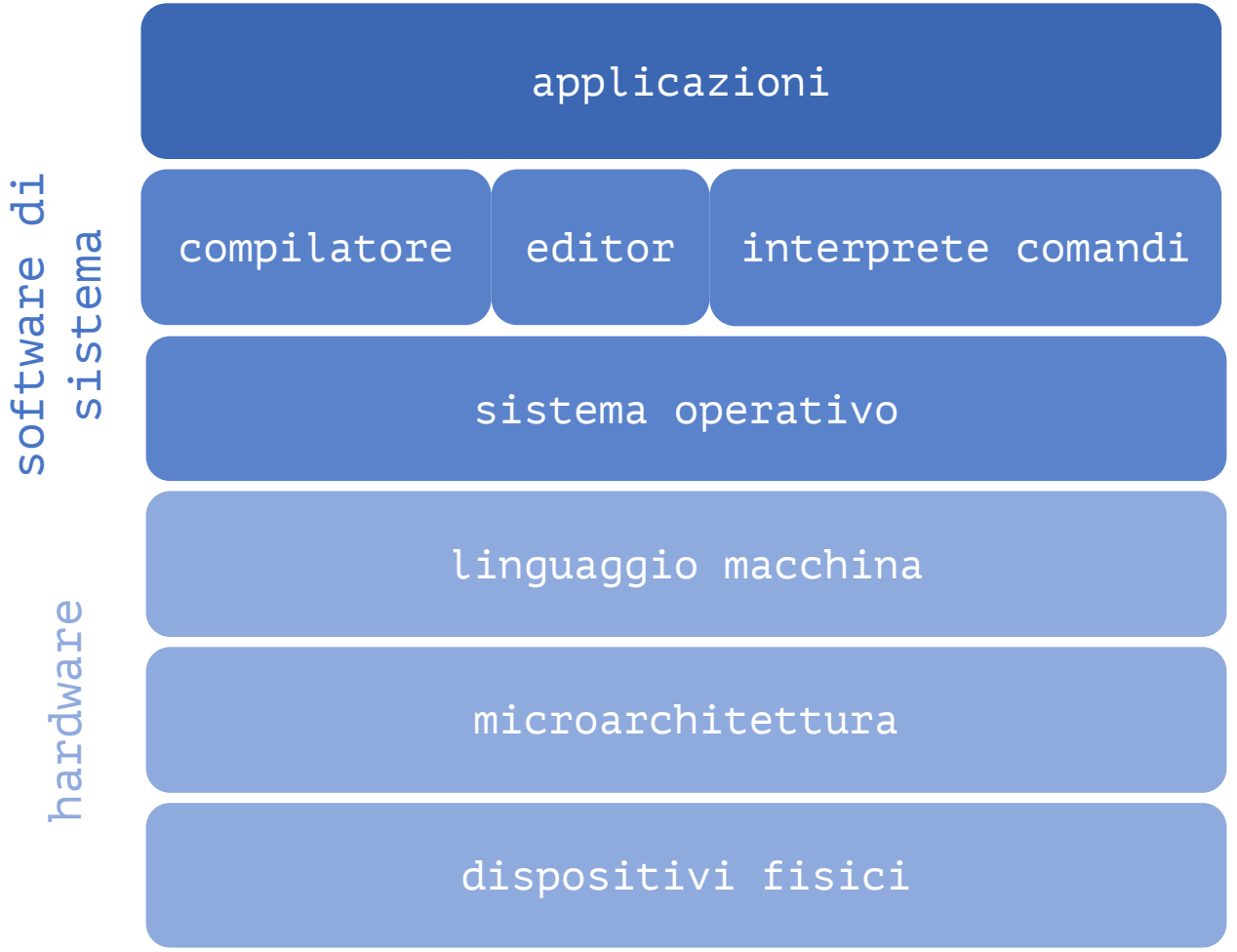
- il calcolatore è un sistema complesso uno (o più) processore, memoria di lavoro, memoria secondaria, periferiche
- l'utente (programmatore) vuole poter scrivere un programma che utilizzi le risorse senza doversi preoccupare di
  - saperle utilizzare (vedendole come scatole nere),
  - gestirle e/o
  - sapere se sono a sufficienza

introduzione

- è necessario uno strato software in grado di utilizzare le specifiche risorse fisiche
  - in relazione alle loro caratteristiche tecniche
  - accedendo alle funzionalità senza dover essere degli esperti
  - anche quando sono quantitativamente limitate
  - senza preoccuparsi di cosa sta avvenendo nel resto del sistema di calcolo

sistema operativo

collezione di programmi (strato software) che maschera la complessità dell'architettura sottostante e fornisce al programmatore un'interfaccia **semplificata** e **potenziata** per l'utilizzo delle risorse



sistema operativo

visione semplificata ed estesa

gestisce le risorse del sistema di calcolo e offre ai programmi utenti la possibilità di accedervi in modo **semplificato**

- mascherando la complessità dell'accesso alla risorsa (lettura di un dato dall'HD non richiede di sapere come avvenga fisicamente questa operazione)
- astraendo i dettagli specifici della risorsa (leggo un dato dalla memoria secondaria ed è una operazione di lettura che prescinde dalla tecnologia e/o caratteristiche del dispositivo)

sistema operativo

visione semplificata ed estesa

offrendo una visione di una macchina **estesa**, **potenziata**

- mascherando il fatto che le risorse possano essere scarse (la memoria di lavoro ha una capacità finita, la stampante in un'aula è unica, condivisa tra tanti PC)
- gestendo i conflitti nell'accesso alle risorse condivise, adottando delle politiche (quando due PC vogliono stampare "contemporaneamente" usa delle priorità)

sistema operativo

gestione risorse

le risorse gestite sono:

- il processore (o processori)
- la memoria di lavoro
- i dispositivi periferici, inclusa la memoria di massa

vengono utilizzati

- **meccanismi** di gestione della risorsa, e
- **politiche** per ottimizzare l'uso della risorsa e risolvere i conflitti d'accesso

sistema operativo

visione semplificata

mette a disposizione degli strati software soprastanti (programmi di sistema, applicazioni, …)
primitive di accesso alle risorse che nascondono i dettagli specifici del dispositivo
(**chiamate di sistema**)

- print (indipendentemente dal tipo di stampante)
- save [file] (indipendentemente dal tipo di supporto SSD, HD, …)

sistema operativo

visione estesa, potenziata

offre agli strati software soprastanti una visione in cui si dispone di **tutte le risorse necessarie** in termini di

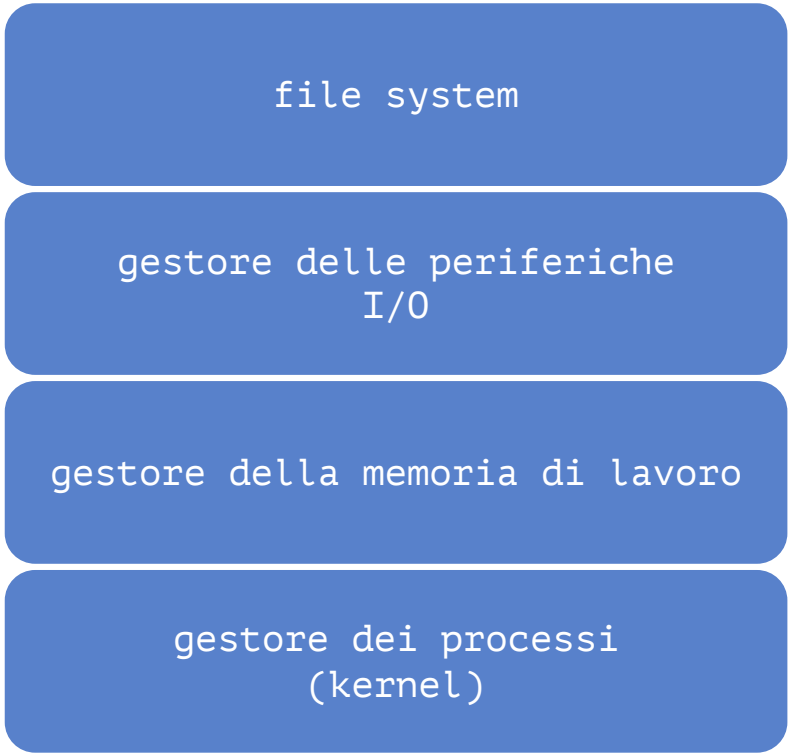
- un processore per l'esecuzione
- la memoria di lavoro necessaria per l'esecuzione
- dispositivi periferici direttamente collegati al sistema principale e pronti alla comunicazione

offre meccanismi di protezione

sistema operativo

collezione di programmi software

- costituito da un insieme di programmi software, ciascuno dei quali
  - gestisce una risorsa del sistema tramite opportuni meccanismi, mascherando un'eventuale limitazione nella quantità della risorsa
  - offre una visione semplificata
  - ottimizza l'accesso alla risorsa per massimizzare le prestazioni adottando politiche di accesso
  - gestisce conflitti nell'accesso da parte di più richieste



chiamate di sistema

- ogni strato del sistema operativo offre agli strati soprastanti le primitive di accesso alla risorsa gestita (chiamate di sistema)
- per accedere alle risorse i programmi effettuano delle chiamate di sistema (chiedono un servizio al sistema operativo)
- il sistema operativo intercetta la chiamata e gestisce la richiesta
  - l'accesso alle risorse è filtrato dal sistema operativo
  - vengono gestiti conflitti, priorità, …
  - offre protezione nell'accesso

gestore dei processi - kernel

- risorsa: processore
- semplifica: per eseguire un programma non è necessario preoccuparsi di caricare le istruzioni in memoria, controllare l'esecuzione contemporanea al funzionamento del calcolatore
- estende: è possibile eseguire qualsiasi numero di programmi contemporaneamente, indipendentemente dal numero di processori a disposizione (considerando che anche il sistema operativo è in esecuzione)
- è come se ogni processo avesse a disposizione un processore dedicato

processo

- programma in esecuzione
  - le istruzioni
  - i dati
  - le informazione relative all'esecuzione - stato (risorse usate, file aperti, posizione in memoria, …)
- le informazioni vengono memorizzate nella tabella dei processi
- l'esecuzione di un programma dà vita a uno o più processi

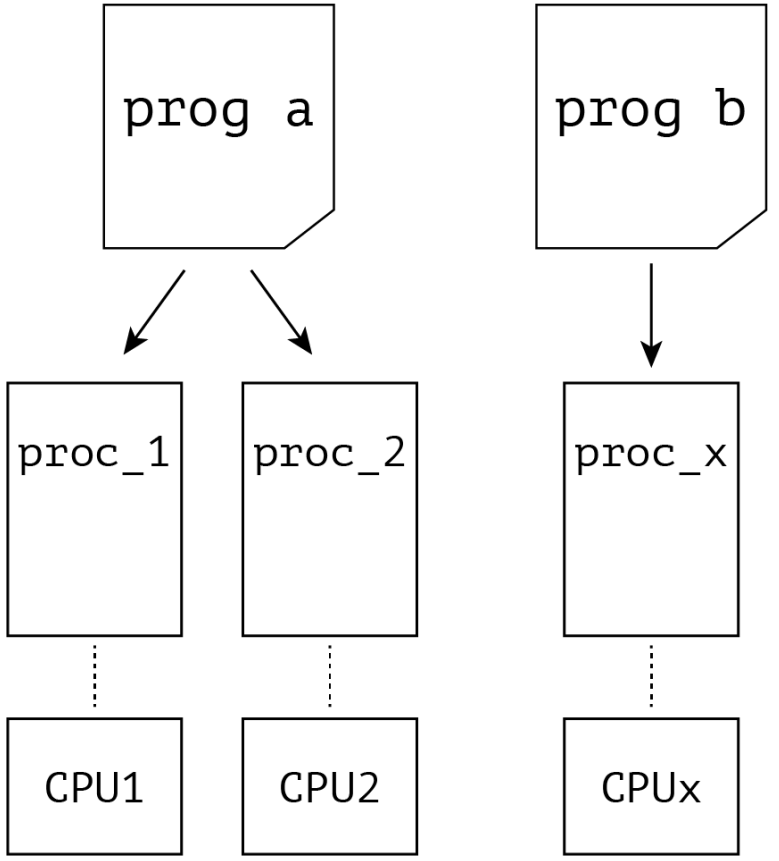
processo

- informazioni sull'utilizzo della memoria
- informazioni statistiche
  - utilizzo del processore
  - limiti di tempo
  - priorità
- informazioni I/O

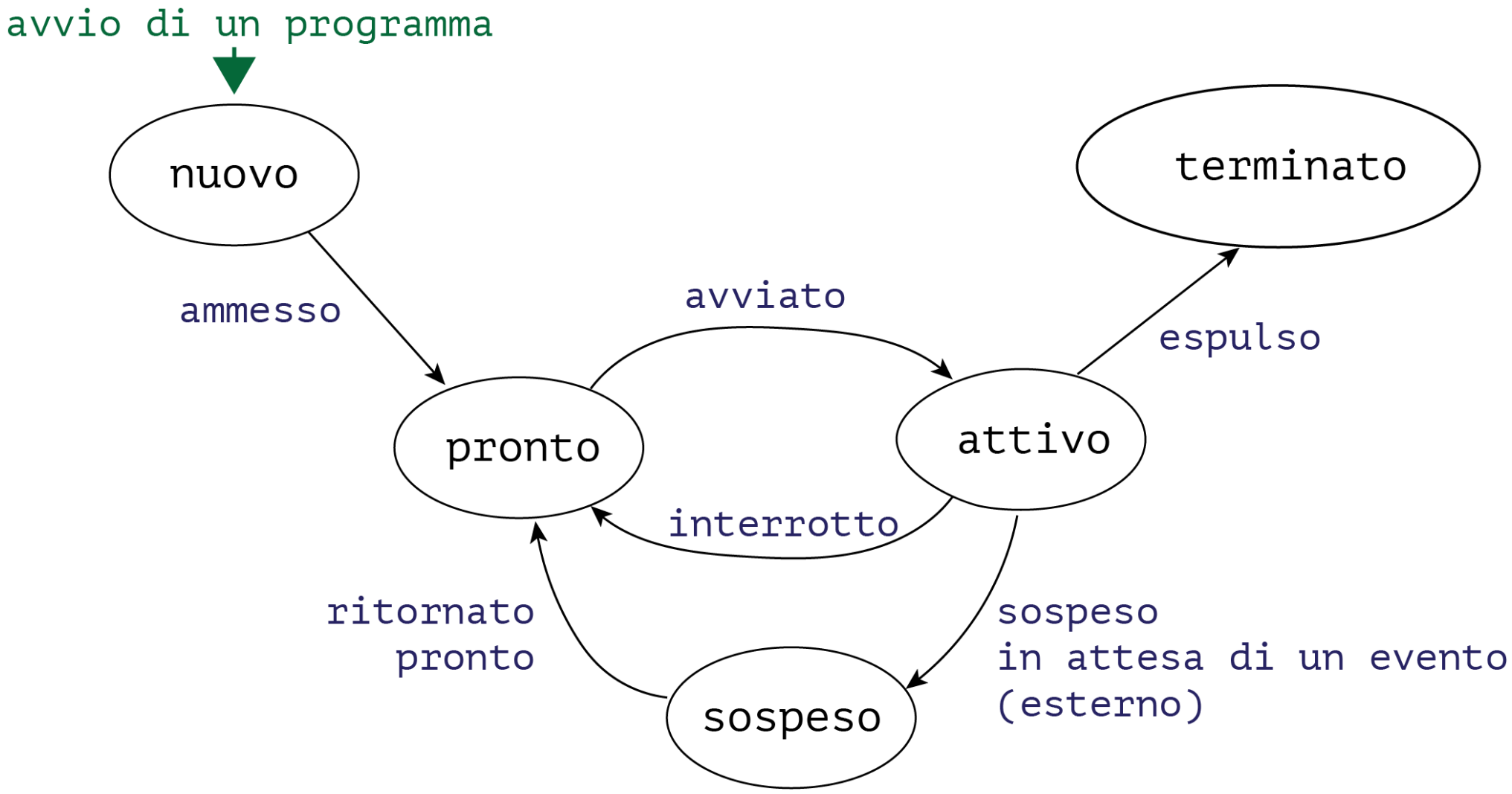
gestore dei processi - kernel

compiti

- realizzare l'esecuzione di un programma
- attività:
  - crea e cancella processi
  - mantiene la tabella dei processi
  - stabilisce quale/i processi assegnare al/ai processori
  - fornisce meccanismi di comunicazione tra processi di uno stesso programma



stati di un processo e transizioni



stati di un processo

- **nuovo** (new): il processo viene creato in seguito ad una richiesta. Una volta inserito nella tabella dei processi con i suoi dati, il processo è pronto per essere eseguito
- **pronto** (ready): il processo può essere eseguito, attende la disponibilità della risorsa CPU
- **attivo** (running): il processo è attualmente in esecuzione (ha assegnata la CPU)
- **sospeso** (waiting): il processo è sospeso in attesa che si verifichi un evento di qualche tipo (come il completamento di un'operazione di I/O). L'esecuzione non può proseguire fino a quando non si verifica l'evento.
- **terminato** (terminated): il processo ha terminato l'esecuzione, ma non è stato ancora fisicamente cancellato dalla memoria di lavoro

transizioni

- nuovo ➤ pronto: ci sono tutte le informazioni necessarie per poter assegnare il processo alla CPU
- pronto ➤ attivo: viene assegnata la CPU al processo
- attivo ➤ pronto: l'esecuzione potrebbe proseguire ma si sceglie di assegnare la CPU ad un altro processo
- attivo ➤ sospeso: l'esecuzione non può proseguire in attesa di una risorsa/evento esterno
- sospeso ➤ pronto: l'esecuzione del processo può riprendere
- attivo ➤ terminato: l'esecuzione è terminata

sistema multiprogrammato

- è possibile avviare più programmi contemporaneamente, indipendentemente dal numero di processori a disposizione
- in ogni istante c'è un processo attivo per processore
- l'utente del sistema ha la percezione che ci siano più processi attivi contemporaneamente: pseudo-parallelismo
- il kernel
  - attua il meccanismo per consentire l'esistenza di più processi contemporaneamente, ad esempio il time sharing
  - adotta politiche di assegnamento (*scheduling*) della CPU ai processi che ne fanno richiesta: round-robin, first-come-first-served, …

kernel: creazione nuovo processo

eventi:

- inizializzazione del sistema
- chiamata di sistema da parte di un altro processo
- esecuzione di un programma da parte dell'utente

operazioni:

- il kernel assegna un identificativo univoco al processo, chiede la memoria necessaria e inizializza le informazioni necessarie al controllo del processo

kernel: terminazione di un processo

eventi:

- completamento dell'esecuzione
- terminazione anticipata a causa di un errore
- terminazione forzata richiesta da un altro processso

operazioni:

- il processo esegue l'ultima istruzione e restituisce il valore d'uscita al kernel
- il kernel rilascia le risorse assegnate al processo (ed elimina le informazioni dalla tabella dei processi)

kernel: pianificazione esecuzione

scheduling

- più processi pronti per essere eseguiti: ogni CPU può eseguire solo un processo alla volta
- meccanismo adottato per il pseudo-parallelismo: parallelismo a livello di programma - **time sharing**
  - ogni processo è eseguito per un intervallo di tempo (**quanto**) poi viene interrotto e la CPU viene assegnata ad un altro processo
- politica di scheduling: determina quale processo mandare in esecuzione e per quanto tempo
  - **round-robin**: ad ogni processo viene assegnato lo stesso quanto di tempo, a rotazione, creando una coda circolare

kernel: pianificazione esecuzione

scheduling

- funzionamento:
  - quando scade il quanto di tempo il processo viene interrotto
  - lo stato del processo viene salvato
  - il processo viene messo in coda ai processi nello stato di pronto
  - si seleziona il primo processo nello stato di pronto
  - si ripristina lo stato precedentemente salvato (se esiste)
  - gli si assegna la CPU

kernel: pianificazione esecuzione

- nel caso un processo abbia bisogno di risorse/eventi non disponibili il processo viene sospeso e posto nello stato sospeso, in attesa dell'evento

kernel: comunicazione tra processi

il kernel realizza anche la comunicazione tra processi per:

- scambio di informazioni
- sincronizzazione

- meccanismi:
  - scambio di messaggi
  - area di memoria condivisa

gestore della memoria di lavoro

- risorsa: memoria di lavoro
- semplifica: per eseguire un programma non è necessario sapere dove sono fisicamente caricati dati ed istruzioni, e non è necessario preoccuparsi di proteggere l'accesso ai propri dati/istruzioni
- estende: è possibile eseguire un qualsiasi programma indipendentemente dalla quantità di memoria necessaria e dal numero di programmi in esecuzione
- è come se ogni processo avesse a disposizione tutta la memoria necessaria

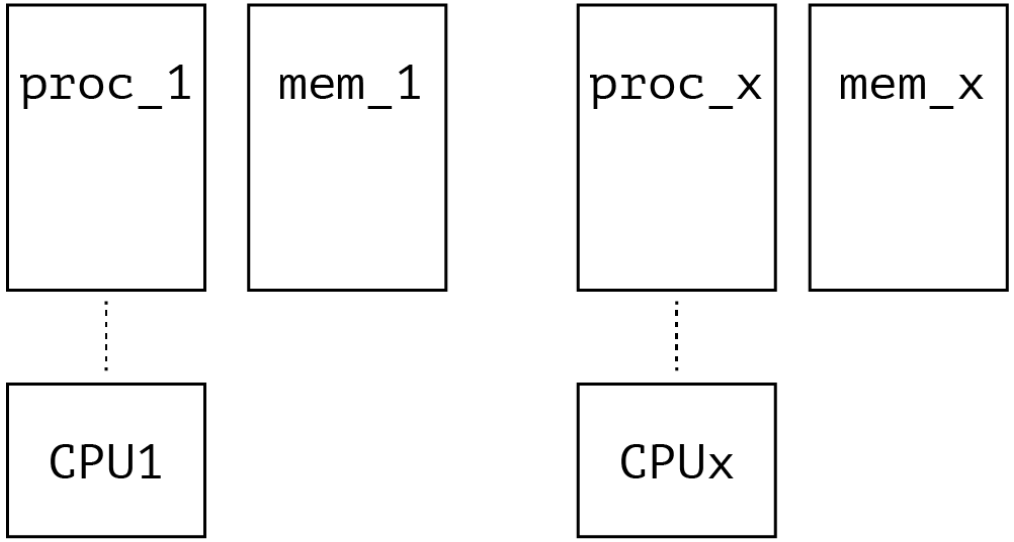
memoria di lavoro potenziata

- quando il numero di indirizzi logici è maggiore del numero di indirizzi fisici non c'è sufficiente memoria di lavoro per tutte le pagine/segmenti
- una parte della memoria secondaria viene utilizzata per salvare le pagine dei processi in stato di pronto o sospeso

- lo spostamento di pagine tra memoria di lavoro e memoria di massa: **swapping**

gestore della memoria di lavoro

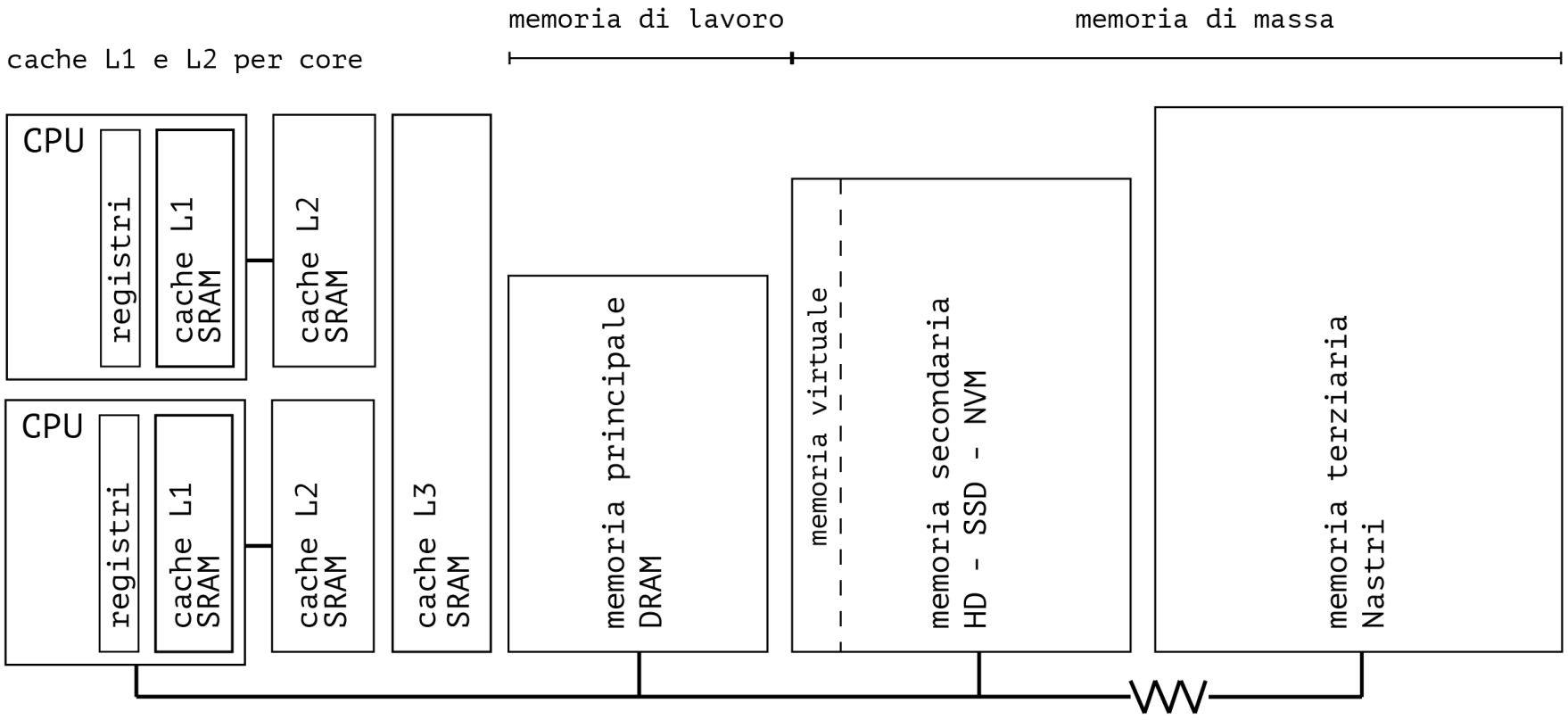
- visione offerta:
  - memoria di capacità infinita,
  - veloce, e
  - non volatile
- nella realtà:
  - memoria insufficiente per tutti i processi attivi
  - memoria insufficiente per un solo processo enorme



gerarchia di memoria

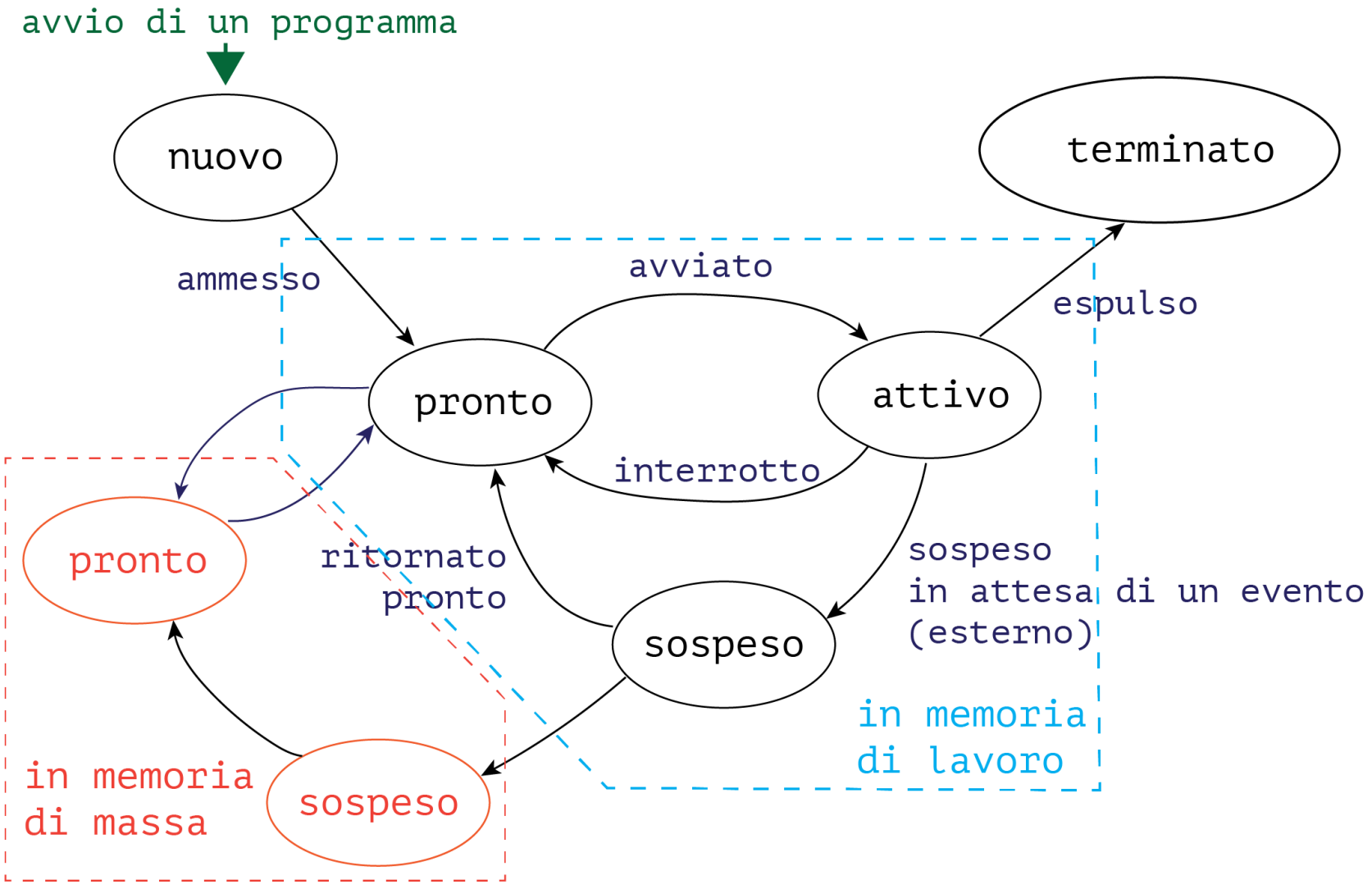
- si introduce la memoria **cache** perché più veloce della memoria principale
  - sfrutta la velocità della memoria cache e le dimensioni della memoria principale
- si usa il meccanismo della **memoria virtuale** per tenere parte delle informazioni dei programmi in esecuzione stanno su memoria di massa (secondaria)
  - sfrutta la velocità della memoria principale e le dimensioni della memoria secondaria

gerarchia di memoria



tempi di accesso (ns)	1	~5	~50	~100	~10000 - ~100000	~10000000000
capacità (KB)	32	256/512	>=8M	>=4GB	>=256GB	TB
costo (€/b)						

stati dei processi e memoria



gestione ed accesso

gestore della memoria di lavoro

- mascheramento della posizione fisica del dato cui si accede tramite il concetto di indirizzo logico
  - consente di poter far riferimento nei processi (programmi) ad una posizione relativa ad un punto di inizio, ignorando quale spazio di memoria verrà effettivamente assegnato al processo
- protezione della memoria di un processo in modo tale che nessun altro processo (non parente di quello proprietario) possa farvi accesso
- si avvale di un dispositivo hardware: Memory Management Unit (MMU)

gestione ed accesso

gestore della memoria di lavoro

- meccanismi per la gestione dello spazio e l'accesso
  - **rilocazione**: conversione degli indirizzi logici utilizzati nel processo in indirizzi fisici relativi all'effettiva posizione in memoria
  - **protezione**: controllo degli accessi alla memoria in base agli identificativi dei processi
  - **paginazione/segmentazione**: suddivisione della memoria in pagine di dimensione fissata da assegnare ai processi - si frammenta il processo in pagine e non è necessario avere tutte le pagine di un processo in memoria per la sua esecuzione

gestione ed accesso

rilocazione

- meccanismo che permette di compilare un programma assegnando indirizzi che presumono che il programma verrà caricato in memoria a partire da un indirizzo noto
  - quando viene effettivamente eseguito, viene posto in memoria in un'area libera, e tutti gli indirizzi vengono "ricalcolati" in base alla posizione reale

- spostamento dei dati di un processo dalla memoria di lavoro alla memoria secondaria e viceversa: swapping

gestione ed accesso

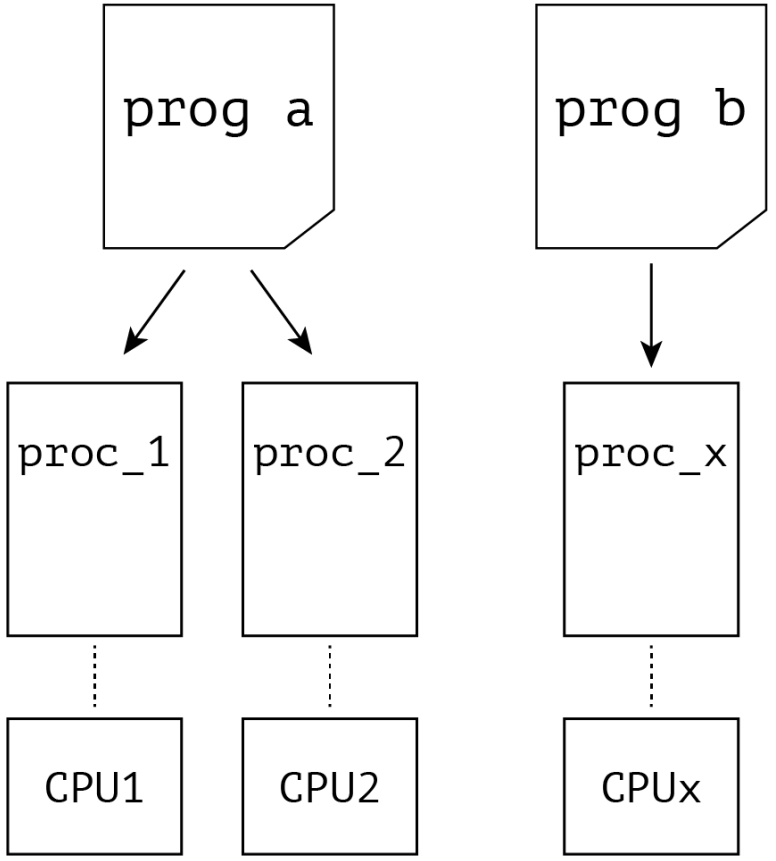
paginazione/segmentazione

- suddivisione dello spazio di memoria in aree di dimensione fissa
  - i processi sono costituiti da una o più pagine
  - non serve poter caricare tutte le pagine di un processo in memoria, solo quella in cui ci sono le istruzioni/dati attualmente usati
- paginazione: tutte le aree hanno ugual dimensione
- segmentazione: aree di diverse dimensioni

gestore della memoria di lavoro

compiti

- realizzare l'esecuzione di un programma
- attività:
  - crea e cancella processi
  - mantiene la tabella dei processi
  - stabilisce quale/i processi assegnare al/ai processori
  - fornisce meccanismi di comunicazione tra processi di uno stesso programma



gestore delle periferiche

- risorsa: periferiche di ingresso/uscita
- semplifica: ogni periferica è una interfaccia che espone delle primitive (comandi di "alto livello") di utilizzo e nasconde gli aspetti meccanici, elettrici, di sincronizzazione del dispositivo
- estende: supporta la condivisione di periferiche che paiono essere dedicate al programma in esecuzione

- è come se ogni processo avesse i dispositivi dedicati

gestione e accesso

gestore delle periferiche

- invia i comandi ai dispositivi
- intercetta e gestisce le interruzioni
  - blocca l'esecuzione del processo corrente
  - salva lo stato
  - richiama la procedura di gestione dell'interruzione (specifica per il dispositivo che ha sollevato la richiesta)
  - ripristina lo stato salvato e riprende l'esecuzione del processo
- organizza e gestisce code di accesso a risorse condivise
- gestisce gli errori

driver

logici e fisici

- driver fisici: appartengono al lato hardware da cui dipendono fortemente e servono per effettuare le operazioni di lettura e scrittura
- driver logici: applicazioni software appartenenti al sistema operativo
  - espongono primitive (print, open, …) che nascondono le caratteristiche dei dispositivi
  - supportano meccanismi di code (ad es. spooling) in modo da far apparire la presenza di più periferiche
- forniti dai produttori dei dispositivi periferici

file system

gestore della memoria di massa

- risorsa: memoria secondaria/di massa - dati
- semplifica: offre una visione logica dello spazio in cui salvare in modo persistente informazioni, in modo che siano facilmente reperibili, espone primitive di alto livello per l'accesso alle informazioni
- estende: consente di vedere come proprio lo spazio del file system indipendentemente dalla locazione fisica/condivisione

file system

- una collezione di file e directory, e operazioni su di essi
- **file**: meccanismo di astrazione per la memorizzazione di informazioni: sequenza di byte
  - nome
  - estensione
  - percorso
  - attributi (ora di creazione, proprietario, ultima modifica, permessi, …)
- **directory**: file speciali di sistema

file system

- organizzazione logica dello spazio di memoria secondaria indipendente
  - dal tipo di tecnologia della memoria
  - dalla posizione fisica (interno al calcolatore, disco di rete condiviso, …)
- offre meccanismi di protezione nell'accesso ai file (proprietario, permessi di lettura/scrittura, …)

riferimenti usati

immagini e spunti presi da

- Carl Hamacher "Computer Organization"
- A. Tanenbaum, H. Bos, "Modern Operating Systems"

corsi presso:

- Stony Brook University
- Carnegie Mellon University