Sistemi di Elaborazioni delle Informazioni I
A.A. 2012/13

Docente: Marco Aldinucci

aldinuc@di.unito.it
Ricevimento: Su appuntamento
http://www.di.unito.it/~aldinuc

Esercitatrice: Alessia Visconti
visconti@di.unito.it

Struttura del corso

- Lezioni in aula (E)
  - Lunedì dalle 14.05 alle ore 16.30
  - Martedì dalle 09.40 alle ore 11.25
  - Mercoledì dalle 11.30 alle 13.05
- Esame
  - Scritto + orale
- Esercitatore
  - Dr. Alessia Visconti

Calendario

28-29-30 Gennaio	Lezione
4-5-6 Febbraio	Esercitazione
11-12-13 Febbraio	Lezione
18-19-20 Febbraio	Lezione
25-26-27 Febbraio	Esercitazione

Programma di massima

- Sistemi Operativi (Tanembaum)
  - Cap. 1 Tutto
  - Cap. 2: 2.1, 2.2.1, 2.3, 2.4.
  - Cap. 3: 3.1 e 3.3
  - Cap. 4: 4.1, 4.3
  - Cap. 5: 5.4, 5.5
  - Cap. 6: 6.1, 6.2, 6.3, 6.4
- Reti di calcolatori (Kurose-Ross)
  - …

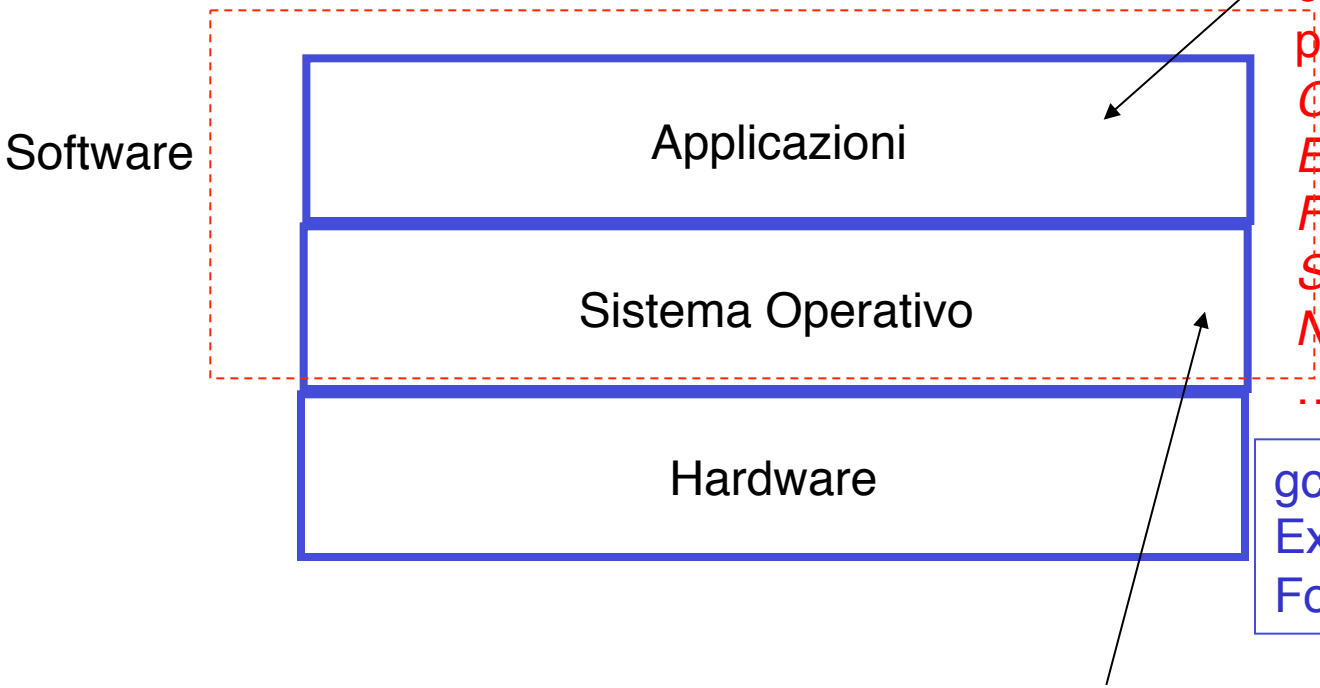
Libri di Testo

- Testo Consultazione
  - A.S. Tanenbaum, *I moderni sistemi operativi (2 ed)*, Jackson 2002

Introduzione

Cos’è un sistema operativo ?

Cos'è un sistema operativo ?



È il livello di SW con cui interagisce l'utente e comprende programmi quali :

Compilatori
Editori di testi
Fogli elettronici
Sistemi bancari
Navigatori per Web
…..

gcc, Word, PowerPoint
Excel, Explorer
Fotoshop, Gimp

Windows
Unix/Linux
Mac OS

È il livello di SW che interagisce direttamente con l'hw e che si occupa di una uso corretto ed efficiente delle risorse fisiche (processore, memorie, periferiche etc.)

Quali sono le funzioni di un SO ?

- Esegue applicazioni :
  - carica il programma binario prodotto della compilazione (e residente su disco) nella RAM,
  - cede il processore all'applicazione da eseguire
- Facilita l'accesso alle periferiche/dispositivi
  - interagisce con le periferiche facendosi carico di tutti i dettagli fisici (es. modem, hard disc, video…)
  - mette a disposizione operazioni di lettura/scrittura, invio/ricezione dati ad alto livello che possono essere usate senza conoscere i dettagli tecnici della periferica

Quali sono le funzioni di un SO ? (2)

- Archivia dati e programmi :
  - mette a disposizione dell'utente una visione **astratta** della memoria secondaria (il file system basato sulle astrazioni : *file/archivi* e *folder/cartelle*)
  - gestisce la realizzazione di queste astrazioni sul supporto fisico (disco) gestendo tutti i dettagli legati alla lettura/scrittura dei settori

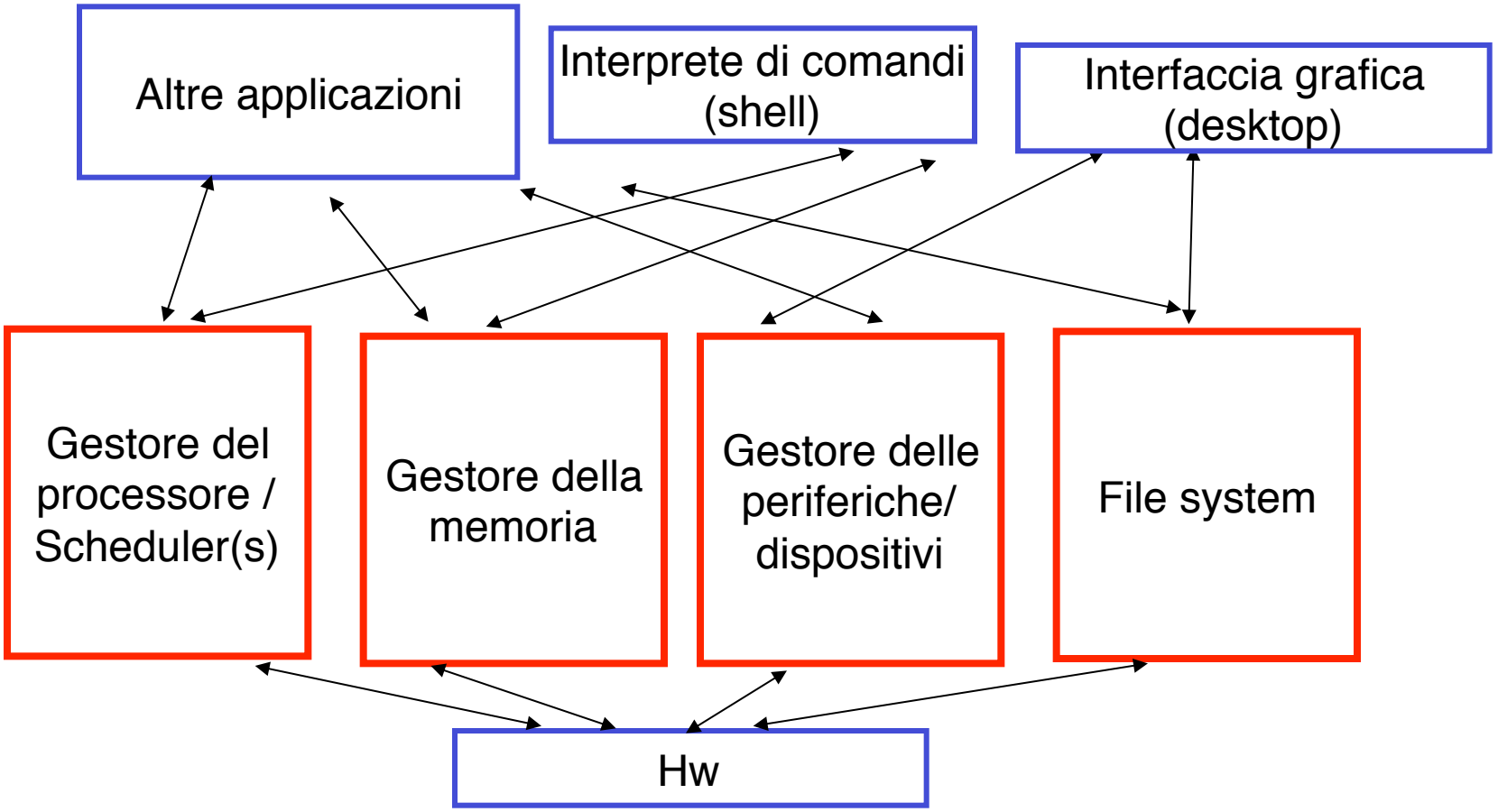
Quali sono le funzioni di un SO ? (3)

- Gestisce le risorse
  - ripartisce le risorse disponibili (processore, RAM, periferiche) fra le varie applicazioni/ utenti
  - evita che ci siano malfunzionamenti dovuti all'uso contemporaneo di risorse
    - es: un word processor e un web browser che inviano contemporaneamente dati alla stampante provocano una stampa erronea
  - ottimizza le prestazioni scegliendo delle politiche che permettano di sfruttare al meglio tutte le parti del computer

Quali sono le funzioni di un SO ? (4)

- Gestisce malfunzionamenti del sistema
  - rileva e gestisce situazioni anomale
    - es: (1) se il disco ha un settore difettoso, il SO può ricopiare le informazioni residenti su quel settore da un'altra parte (in modo trasparente all'utente)
    - es: (2) se un'applicazione cerca di effettuare una operazione non permessa (come leggere i dati di un'altra applicazione) il SO può bloccare l'applicazione segnalando all'utente la situazione erronea

Quali sono le parti di un SO ?

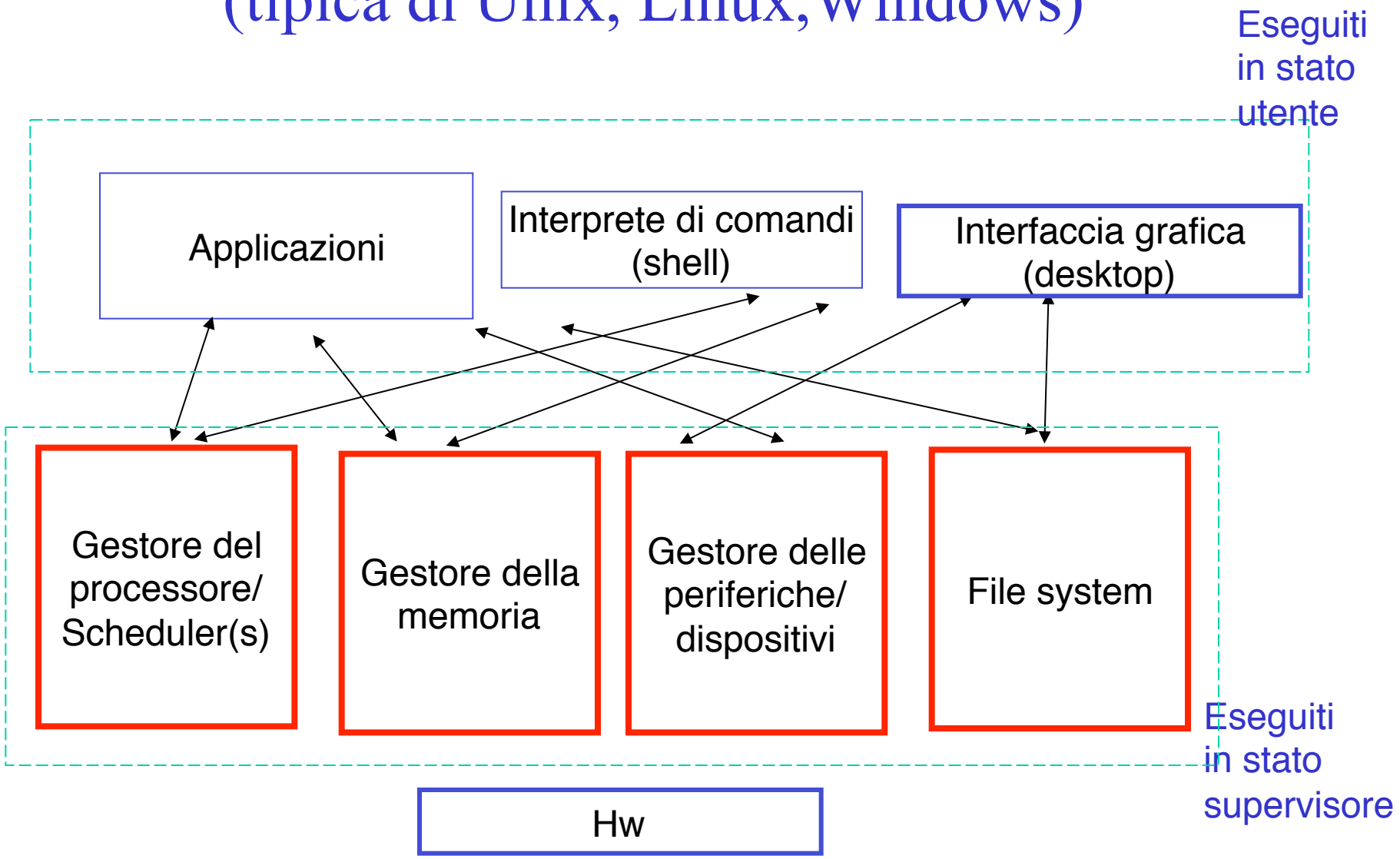


Stato utente e stato supervisore

- Stato utente :
  - modalità di funzionamento dell'hw che permette l'accesso solo a un sottinsieme delle risorse disponibili
    - es : un sottoinsieme delle istruzioni assembler (non si può accedere alle istruzioni che istruiscono le interfacce di I/O), una sola parte della RAM etc.
- Stato supervisore o kernel :
  - modalità che permette l'accesso a tutte le risorse

Organizzazione Monolitica

(tipica di Unix, Linux,Windows)



Organizzazione Monolitica (2)

- I programmi che girano in stato utente *richiedono* servizi al SO tramite invocazione di funzioni 'speciali'
  - *system call* o *chiamate di sistema*
- Le SC portano il sistema in stato kernel e mandano in esecuzione il SO
- Il sistema operativo decide come e quando effettuare il servizio

Organizzazione Monolitica (3)

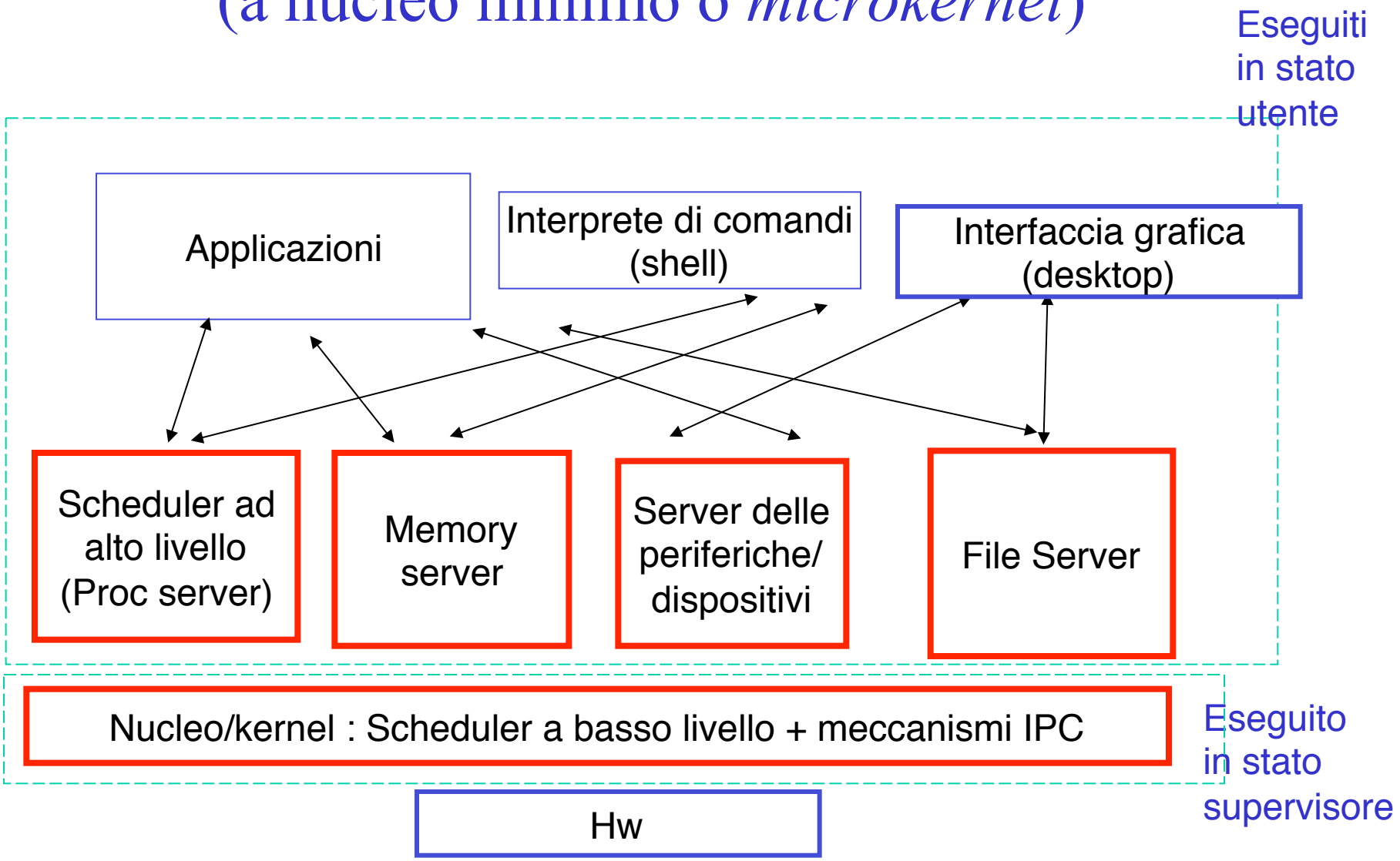
- Il programma utente può essere riattivato con due politiche :
  - alla fine del servizio :
    - si parla di *system call bloccanti* perché l'esecuzione del processo viene bloccata in attesa della fine della gestione della richiesta
  - alla fine della richiesta :
    - quando la richiesta è stata accettata dal SO il processo può continuare a fare altre cose
    - è necessario un meccanismo aggiuntivo per decidere quando la richiesta è stata servita

Organizzazione Monolitica (4)

- Unix, Linux, Windows tipicamente usano SC bloccanti
- Il sistema operativo può *interrompere* l'esecuzione di un programma utente per effettuare operazioni di gestione
  - *questo avviene attraverso il meccanismo delle interruzioni hw*
- Organizzazione del sw del SO :
  - insieme di procedure compilate in un unico oggetto
  - ogni procedura può chiamare tutte le altre/ha visibilità globale

Organizzazione Client-Server

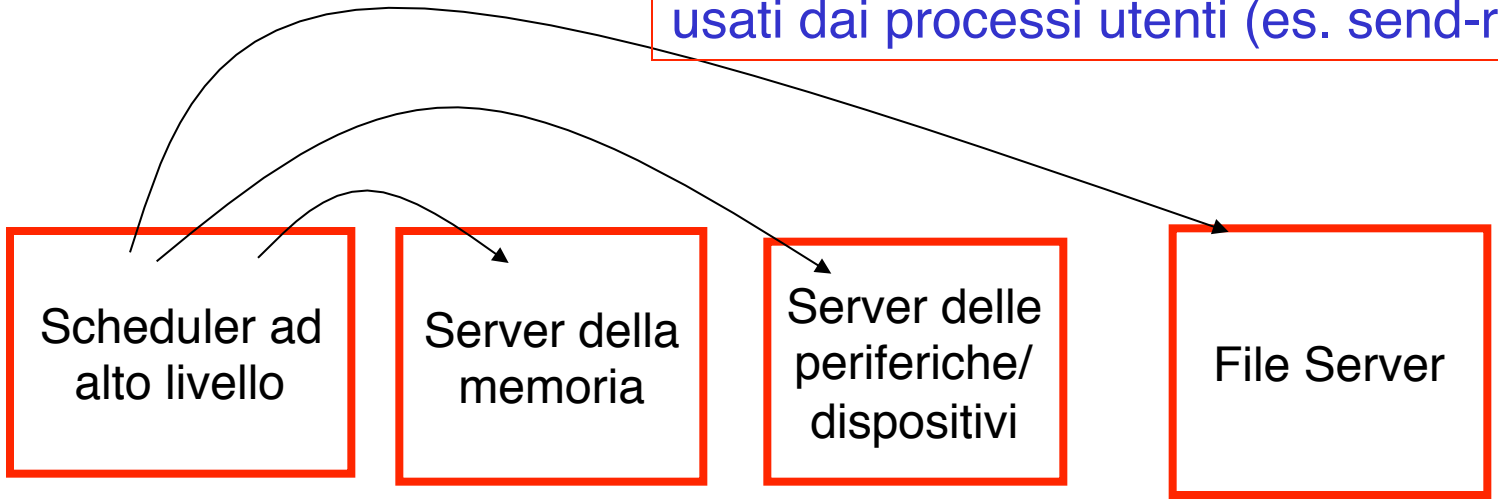
(a nucleo minimo o *microkernel*)



Organizzazione Client-Server (2)

(a nucleo minimo o *microkernel*)

I vari processi server non hanno SD
a comune e comunicano fra di lori con
i normali meccanismi di IPC
usati dai processi utenti (es. send-receive)



Organizzazione Client-Server (3)

- Minimizza le funzioni del SO che girano in modo kernel
- Molte funzioni sono realizzate da processi server che girano in modo utente
- Nucleo minimo (Microkernel) :
  - funzioni base per la gestione dei processi e comunicazione fra processi (IPC)
  - comunicazione con i dispositivi vista come messaggi “speciali”

Organizzazione Client-Server (4)

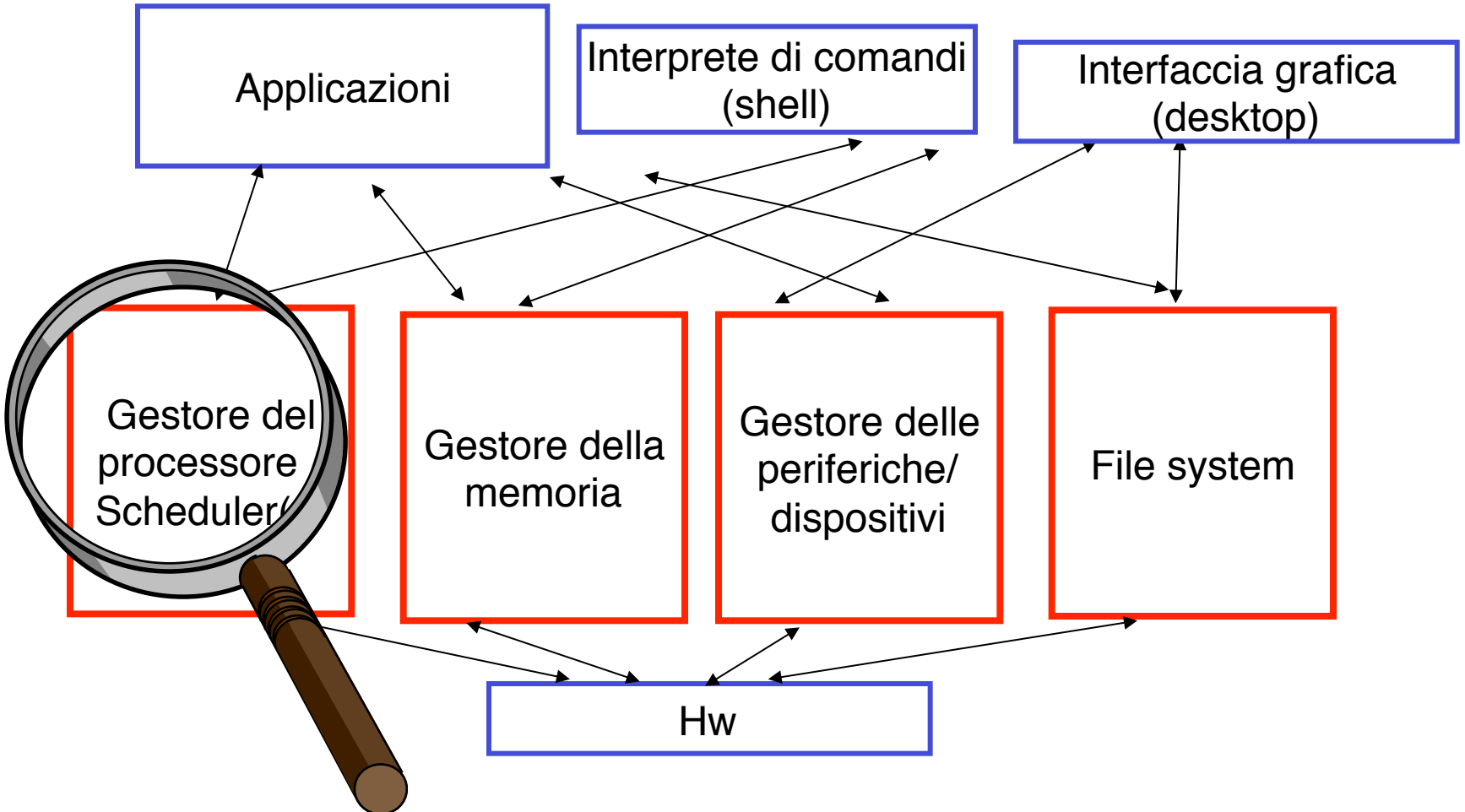
- Quando un processo richiede un servizio *comunica* con uno dei processi server
  - es: effettua una *send* al file server per richiedere la lettura da un file
- L'attesa della terminazione di una servizio avviene come *attesa di una comunicazione*
  - es: effettua una *receive* al file server per ottenere le informazioni lette

Client-server *vs* modello monolitico

- Più sicuro
- Meno efficiente
- Si adatta bene ai sistemi operativi di rete
- Windows NT 3.0 adottava un modello ispirato al client/server (ibrido)
  - scartato perché troppo lento
- Studiato in ambito accademico
  - es: MACH, Minix, sono versioni di Unix a microkernel
  - MacOS (>10.x)

Nel resto del corso ci concentreremo sui sistemi operativi monolitici !

Organizzazione di un SO monolitico

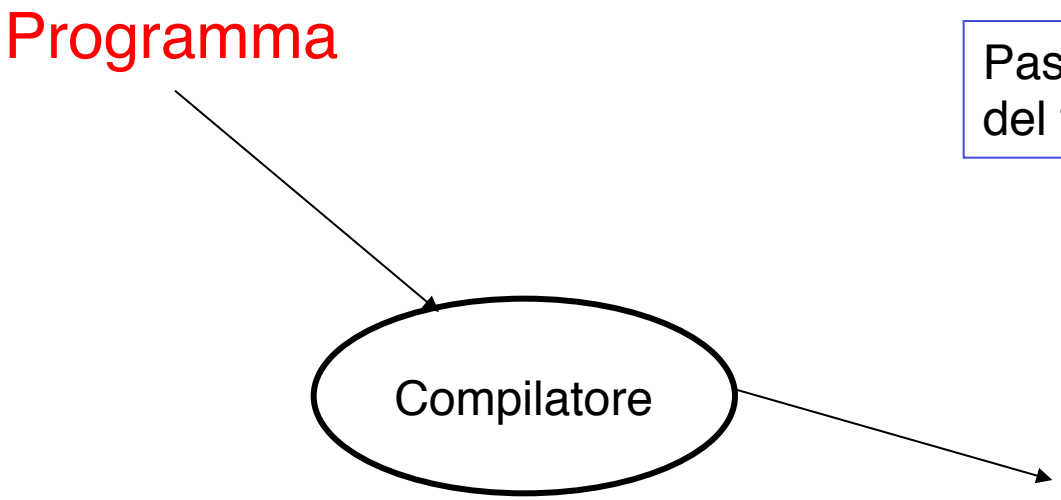


La gestione del processore

Prologo : (1) Come avviene l'esecuzione di un programma ?

(2) Che problemi sorgono se più programmi sono attivi contemporaneamente?

Esecuzione di un programma

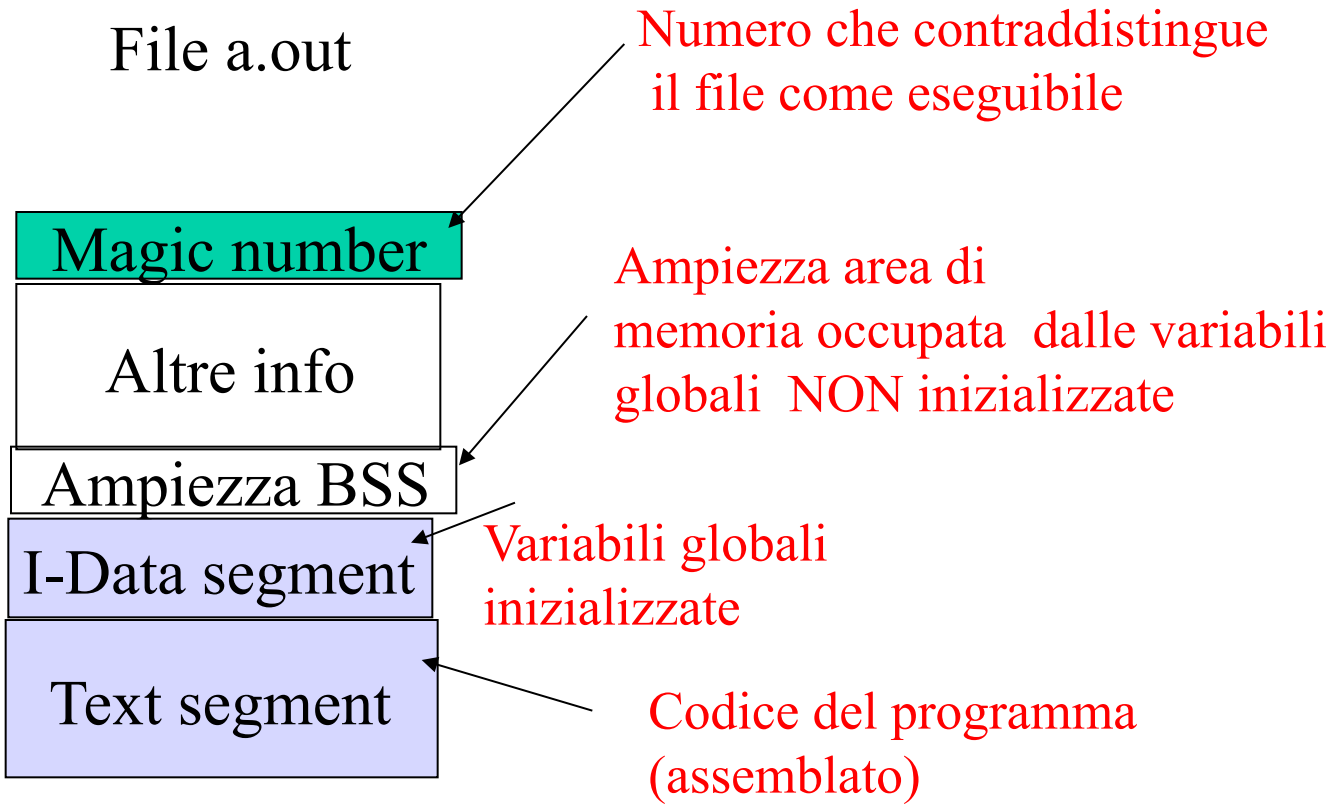


Passo 1 : compilazione e creazione del file eseguibile

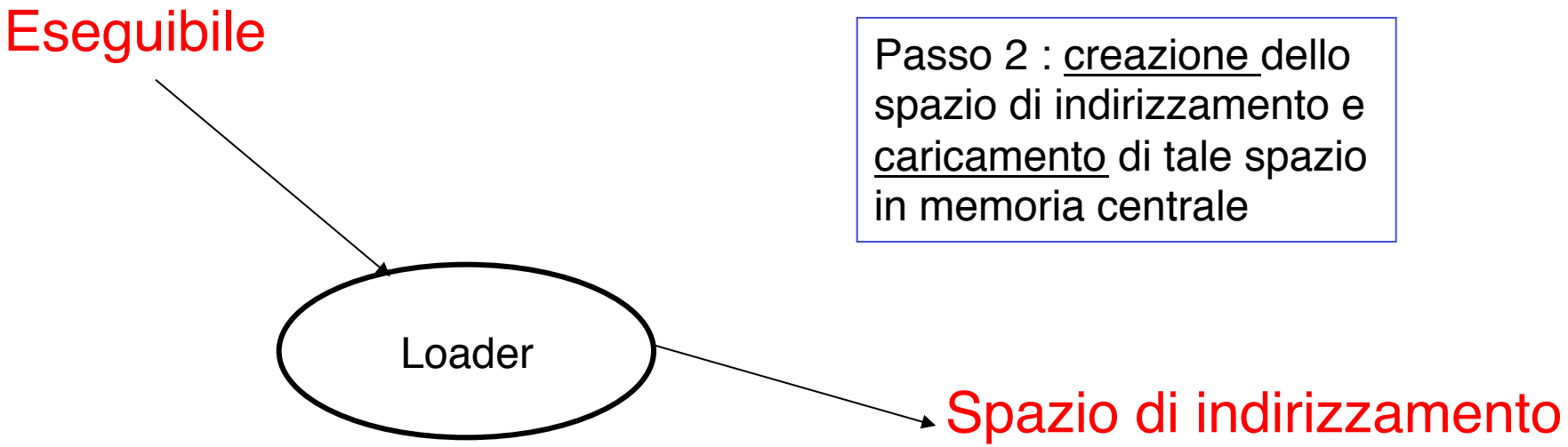
È un file binario che
* contiene tutte le informazioni necessarie all'esecuzione del programma da parte del processore
* ha un formato che dipende dal SO che deve curarne l'esecuzione
* è memorizzato su disco

Formato del file eseguibile

- Un esempio : il formato ELF di Linux



Esecuzione di un programma (2)

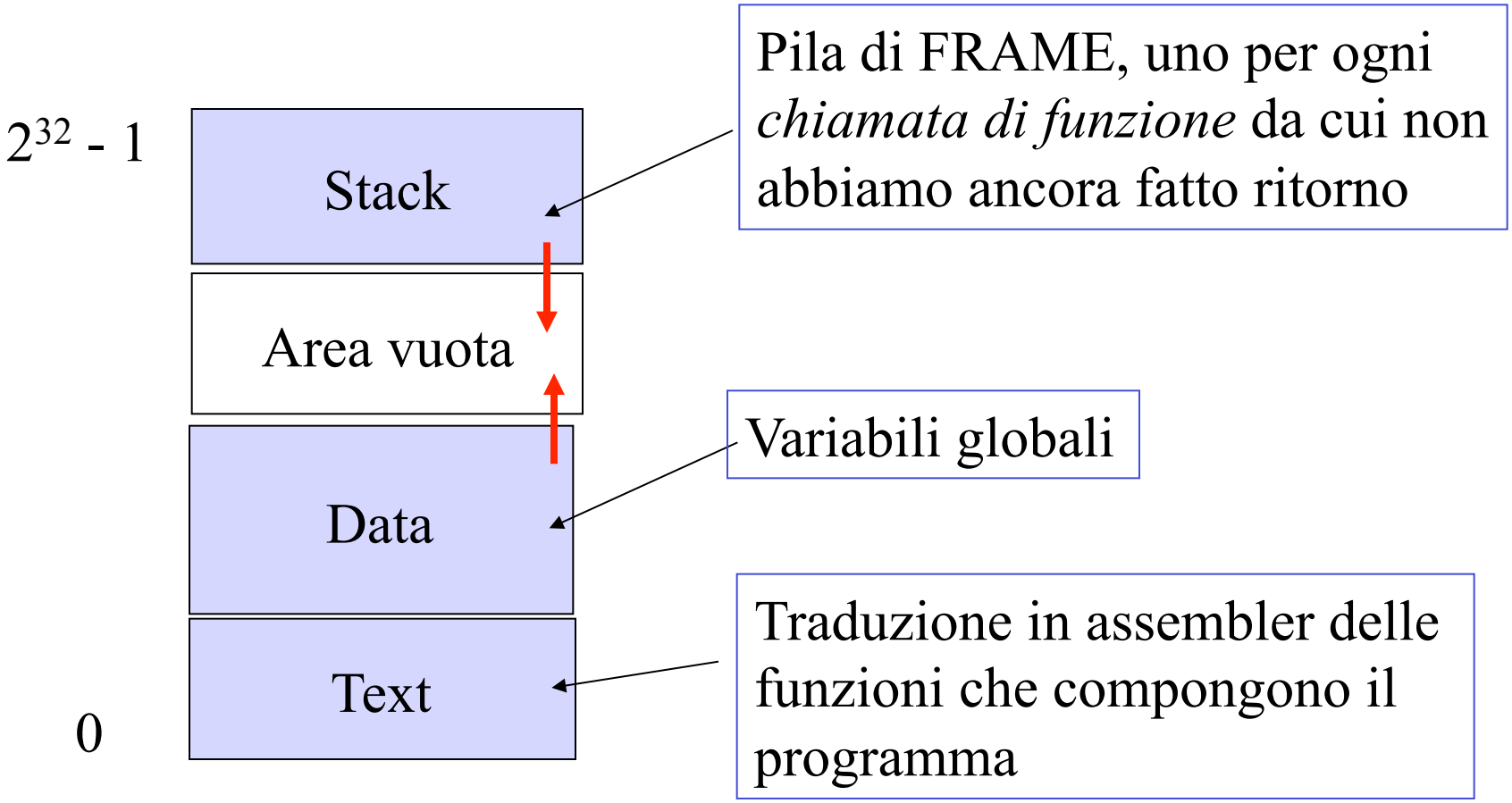


Passo 2 : creazione dello spazio di indirizzamento e caricamento di tale spazio in memoria centrale

È l'immagine della memoria visibile al programma durante la sua esecuzione

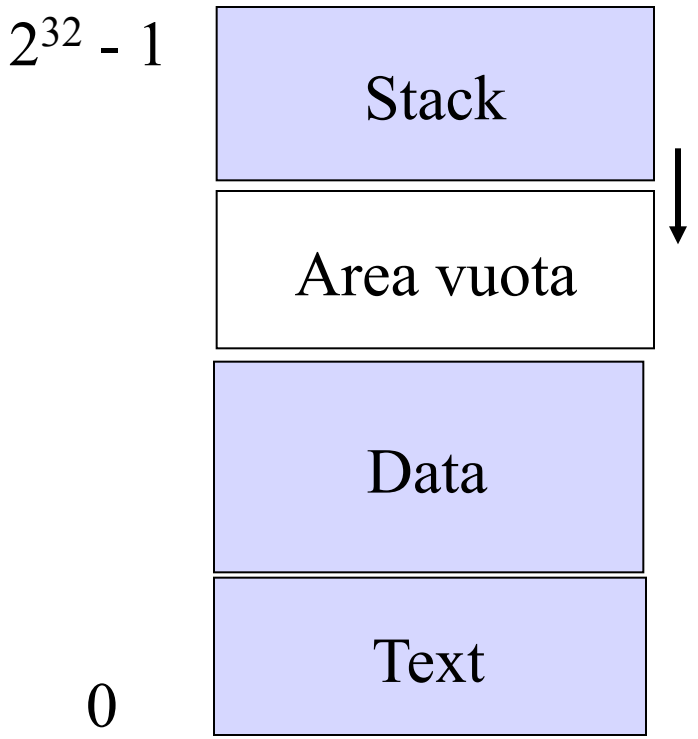
Spazio di indirizzamento

- Come è organizzata la memoria accessibile ad un programma in esecuzione ?



Spazio di indirizzamento (2.1)

- Spazio di indirizzamento tipico (caso del linguaggio C)

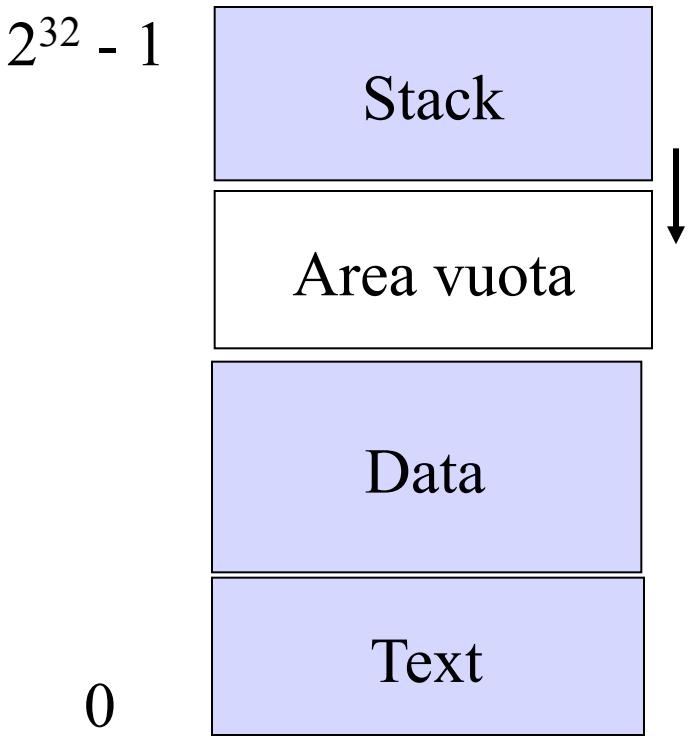


Direzione di
crescita dello stack

Contenuti tipici di un FRAME :
- variabili locali della funzione
- indirizzo di ritorno (indirizzo dell'istruzione successiva a quella che ha effettuato la chiamata alla funzione)
- copia dei parametri di chiamata

Spazio di indirizzamento (2.2)

- Spazio di indirizzamento tipico (caso del linguaggio C)

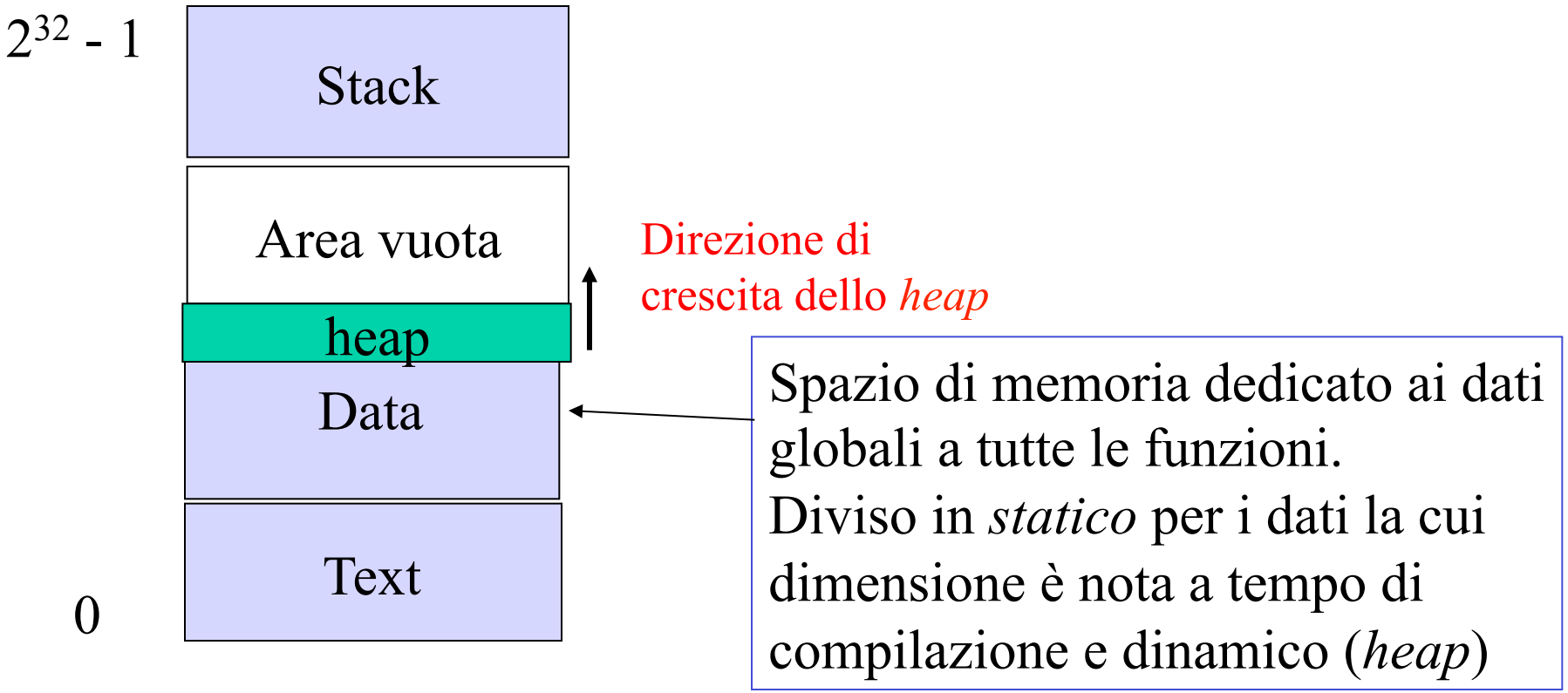


All'inizio dell'esecuzione lo Stack contiene solo il FRAME per la funzione `main`

Successivamente :
* ogni volta che viene chiamata una nuova funzione viene inserito un nuovo frame nello stack
* ogni volta che una funzione termina viene eliminato il frame in cima dello stack e l'esecuzione viene continuata a partire dall'*indirizzo di ritorno*

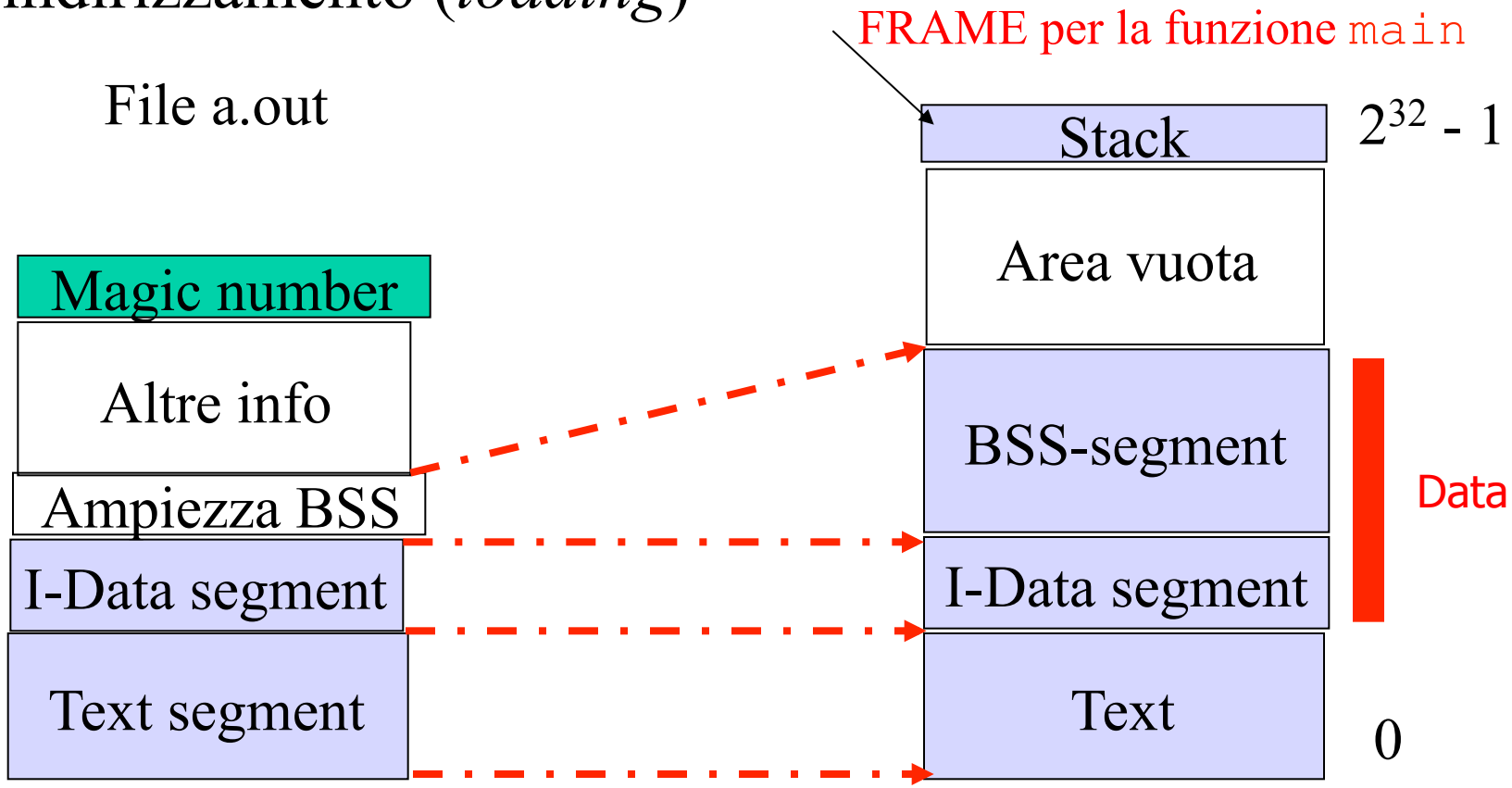
Spazio di indirizzamento (2.3)

- Spazio di indirizzamento tipico (caso del linguaggio C)



Spazio di Indirizzamento (3)

- L'eseguibile contiene tutte le informazioni per creare la configurazione iniziale dello spazio di indirizzamento (*loading*)



Esecuzione di un programma (3)

Passo 3 : attivazione del programma, ovvero
caricamento nel PC dell'indirizzo della prima istruzione
da eseguire nell'area *testo*

A questo punto il programma ha il controllo del processore

Il Sistema Operativo potrà tornare in esecuzione solo
se si verifica uno dei seguenti eventi :
-- arrivo di una interruzione hw
-- terminazione del programma
-- invocazione esplicita di un servizio tramite una System Call

Esecuzione di un programma (4)

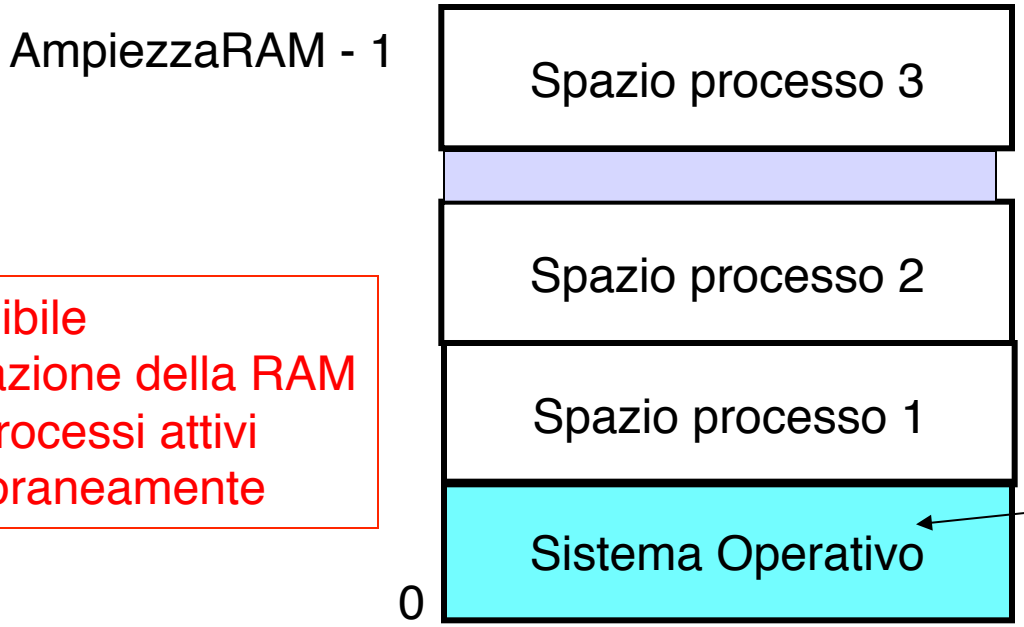
- Processo (Def.) :

  programma in esecuzione completo del suo stato (spazio di indirizzamento, contenuto dei registri, file aperti…)

- Il concetto di processo è centrale nella organizzazione di ogni SO
- Tipicamente ad ogni istante ci sono molti processi attivi contemporaneamente
- I processi *non interattivi* sono anche detti JOB

Condivisione della RAM

- Tipicamente la RAM contiene lo spazio di indirizzamento di più processi :
  - es. organizzazione tipica della memoria negli SO degli anni ‘70 (es IBM 360)

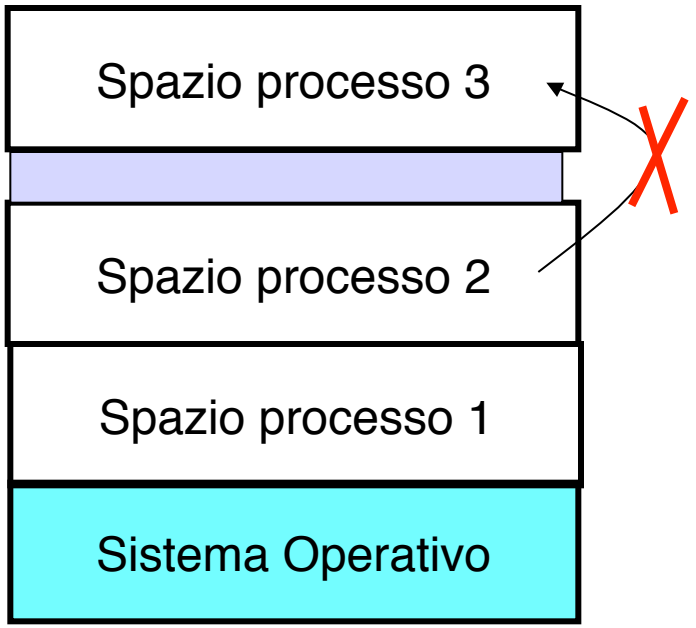


Una possibile organizzazione della RAM con più processi attivi contemporaneamente

Condivisione della RAM (2)

- Problemi legati alla condivisione della RAM :

  (1) protezione dello spazio di indirizzamento di processi diversi



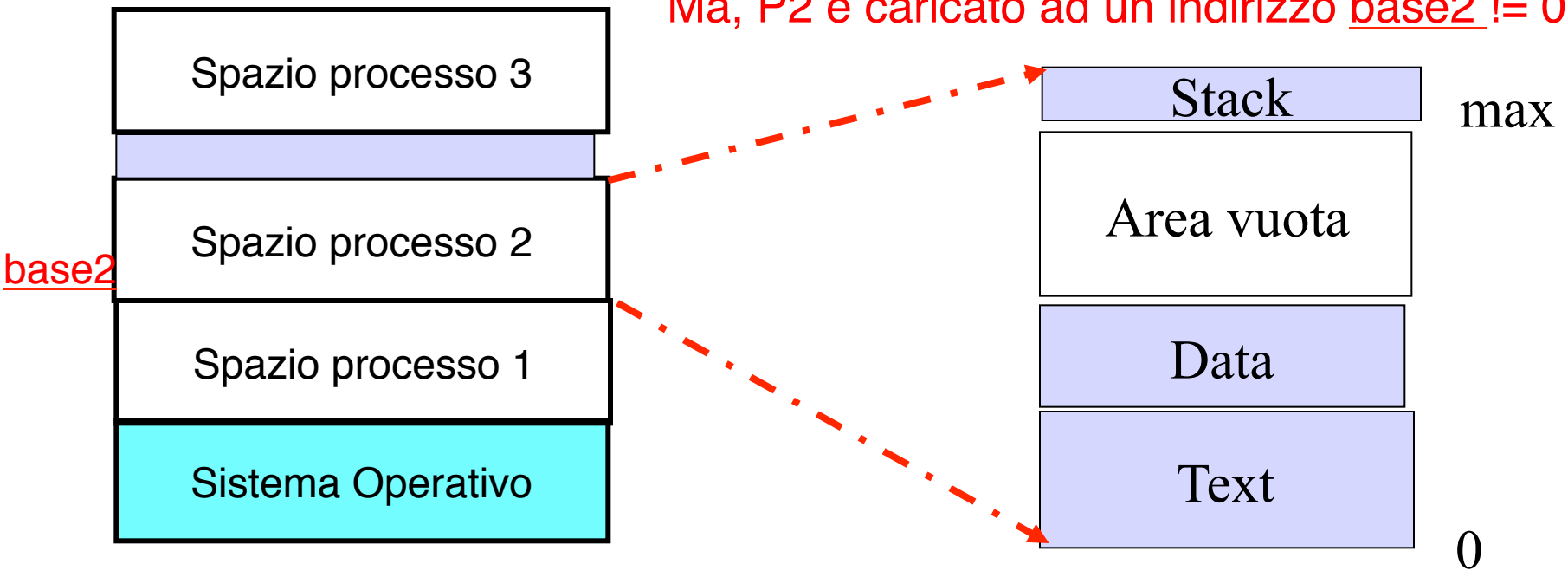
Il processo 2 non deve poter accedere agli indirizzi di RAM al di fuori della sua area

Condivisione della RAM (2)

- Problemi legati alla condivisione della RAM :

  (2) problema della rilocazione

Il compilatore/linker assume che lo spazio di indirizzamento parta dall'indirizzo 0
Ma, P2 è caricato ad un indirizzo base2 != 0

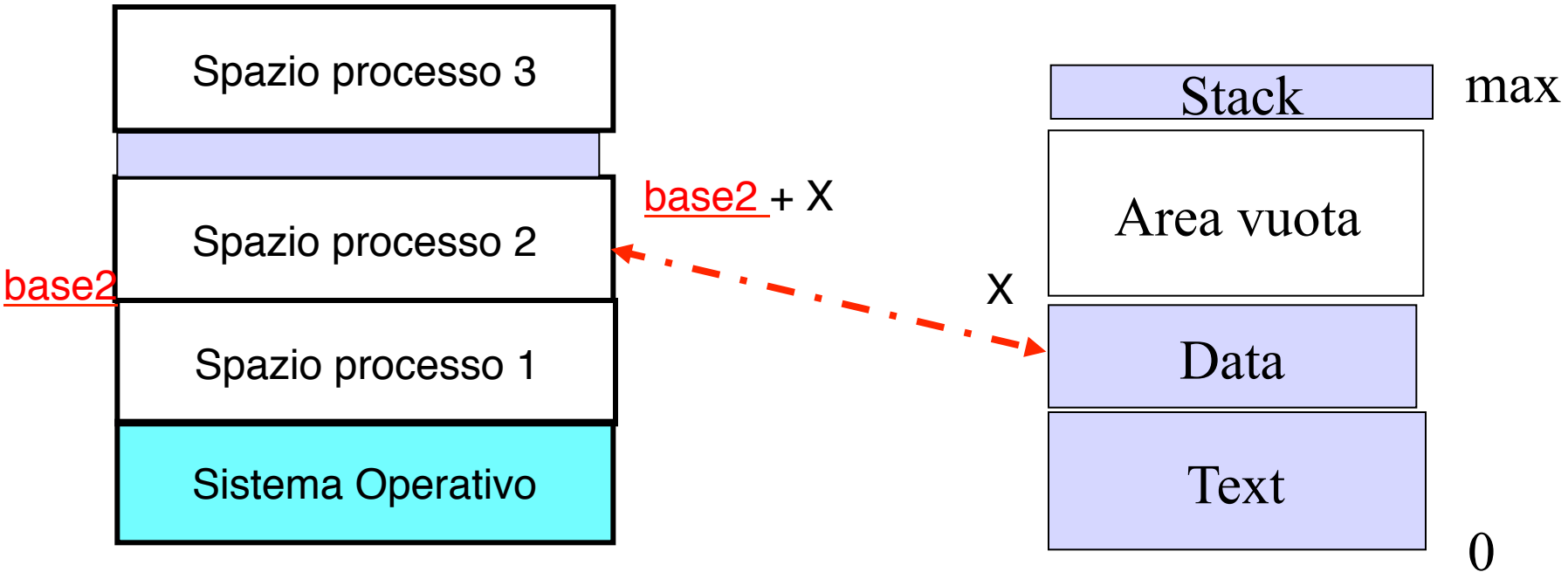


Condivisione della RAM (3)

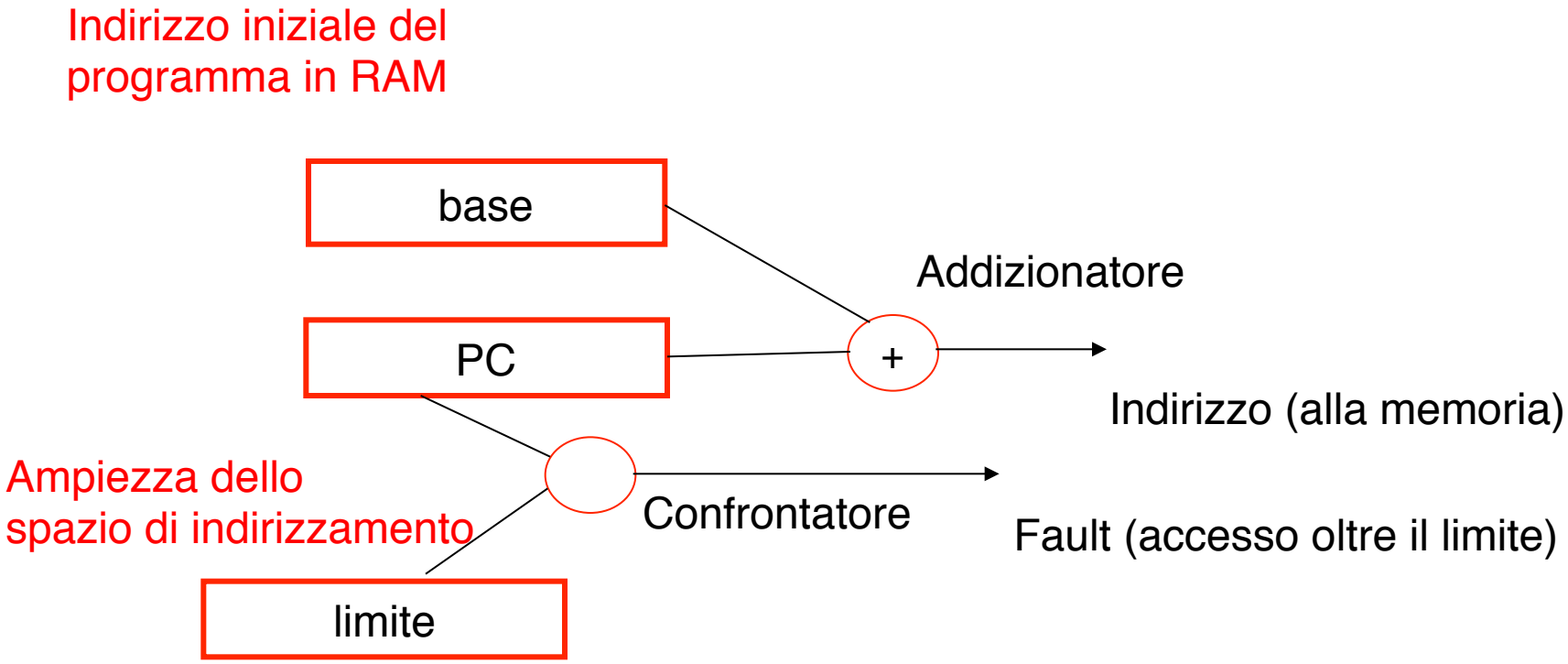
- Problemi legati alla condivisione della RAM :

  (2) problema della rilocazione (cont.)

Indirizzi di istruzioni e dati devono essere incrementati (*rilocazione*) di base2

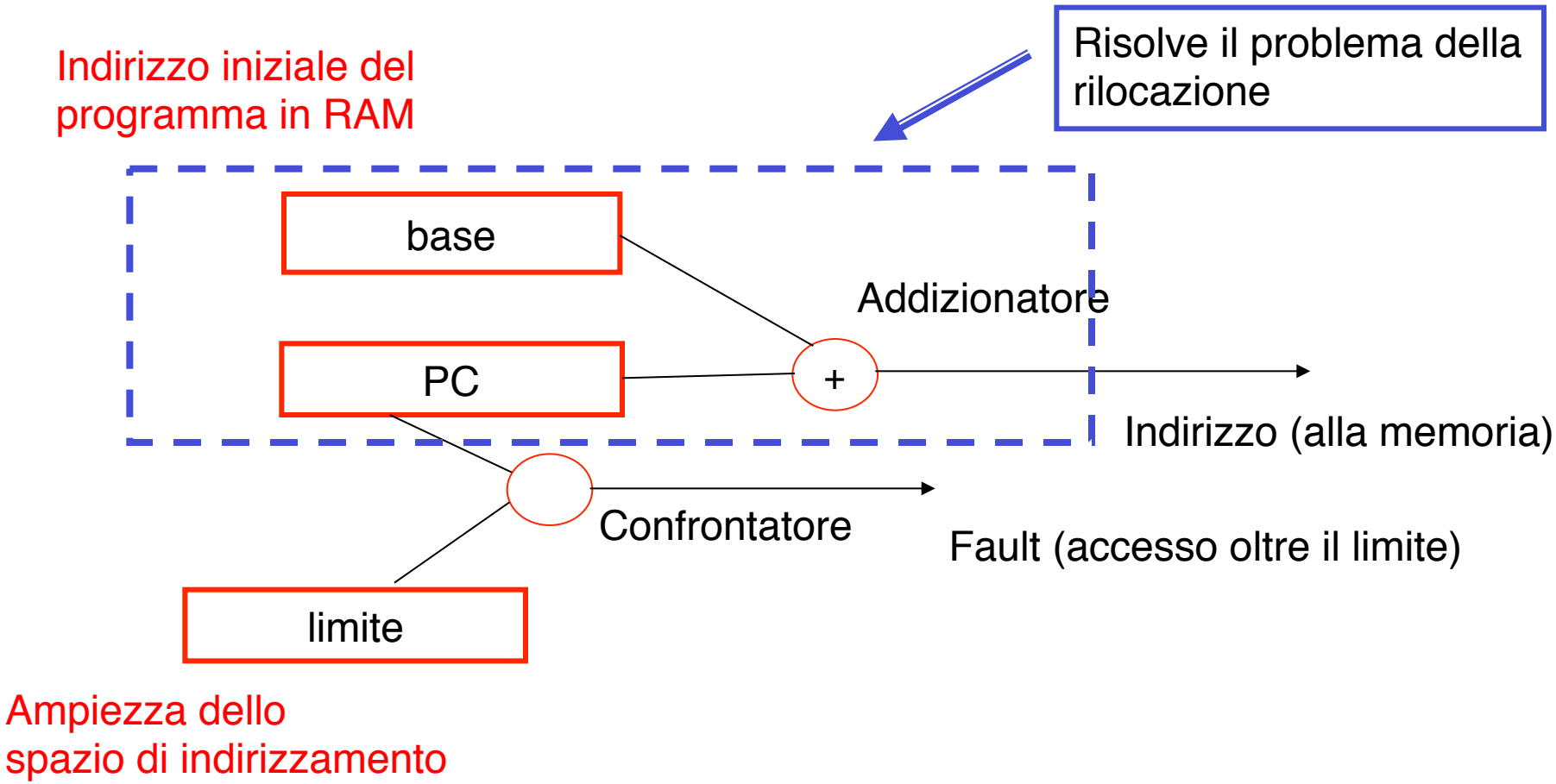


Condivisione della RAM (4)



- È necessario dell'Hw aggiuntivo
- Una possibile soluzione (usata nella serie IBM360)

Condivisione della RAM (5)



Condivisione della RAM (6)