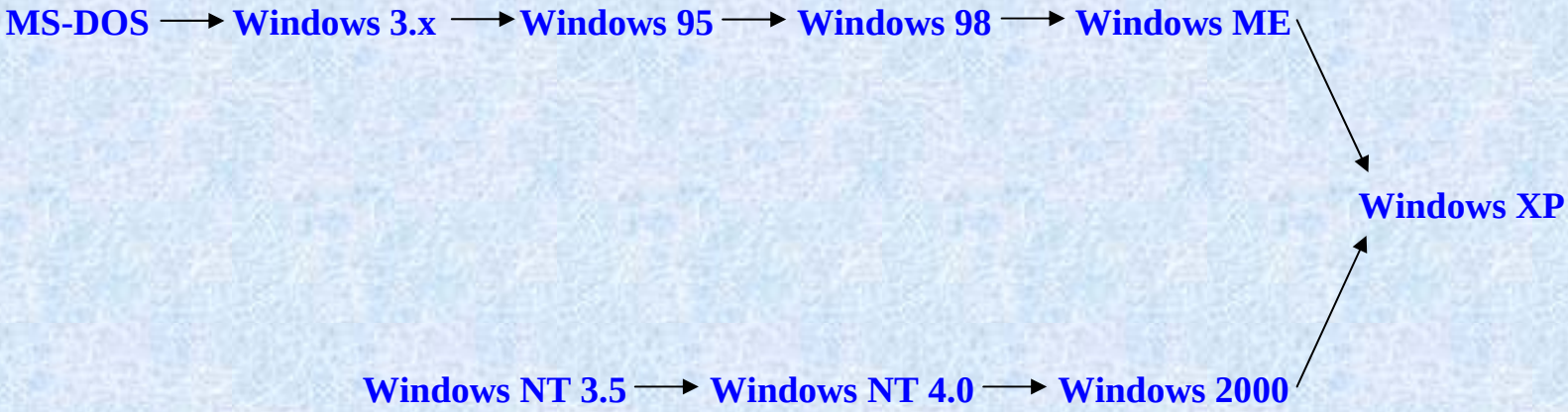
Il sistema operativo Windows

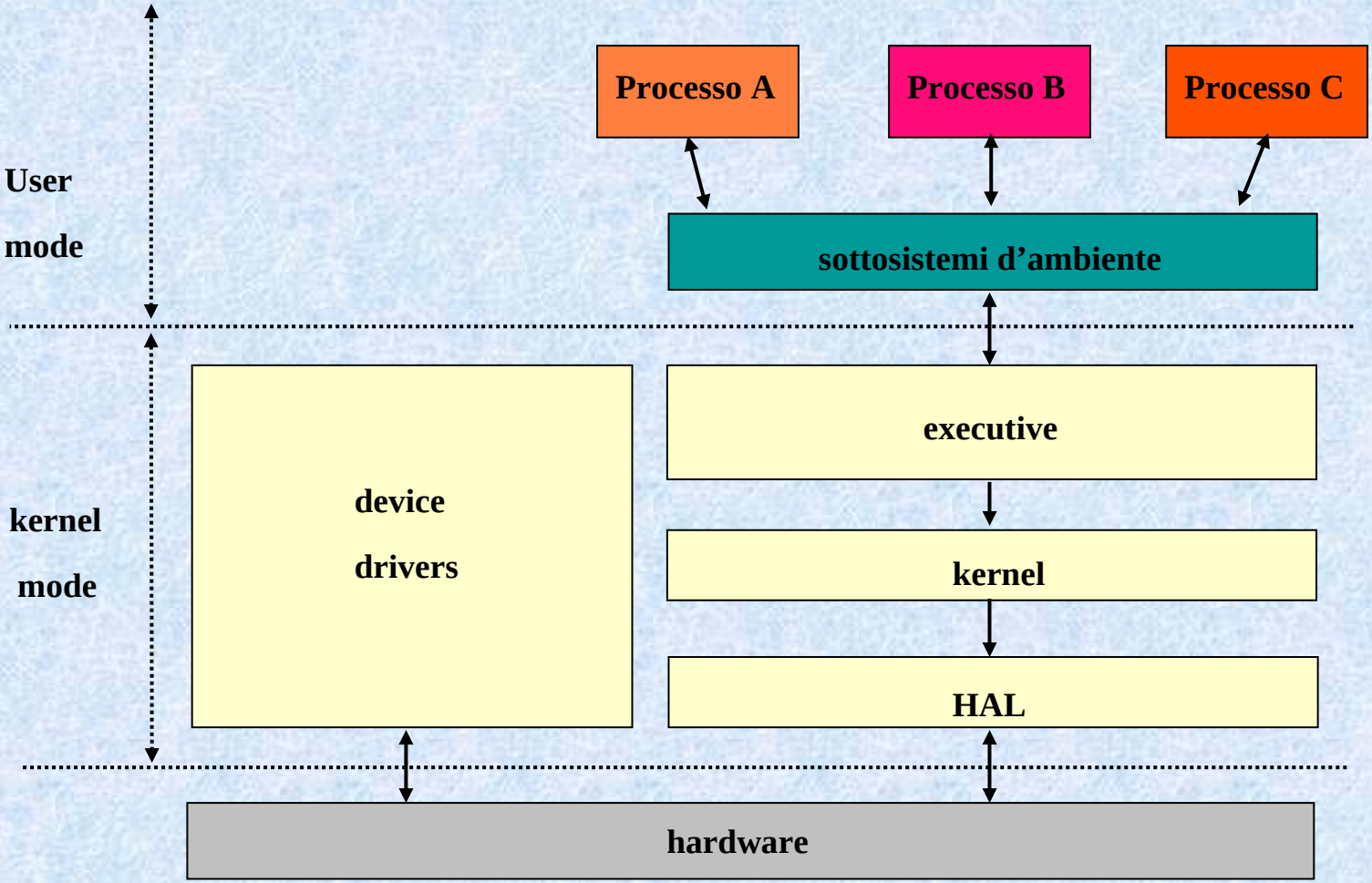
8.1 Struttura generale

- **Evoluzione storica dei sistemi microsoft Windows**



8.1 Struttura generale

- Struttura modulare di Windows



8.2 Gestione dei processi e dei thread

- Un processo in Windows è un “oggetto di nucleo” (*kernel object*) definibile come un "contenitore di risorse”. È caratterizzato da:
  - ✓ Un identificatore
  - ✓ Uno spazio di indirizzamento privato
  - ✓ Una directory corrente
  - ✓ Varie tabelle contenenti le risorse del processo
  - ✓ Uno o più thread di esecuzione

8.2 Gestione dei processi e dei thread

- **Un thread è un "oggetto di nucleo" che definisce una entità concorrente e schedulabile.**
  È caratterizzato da:
  - ✓ **Un identificatore**
  - ✓ **Una funzione da eseguire**
  - ✓ **Un contesto (insieme di registri del processore)**
  - ✓ **Un puntatore allo stack**

8.2 Scheduling dei thread

- Politica di scheduling dei thread: intermedia tra la politica prioritaria e quella “round-robin”
- La priorità di un thread viene calcolata come somma di due componenti:
  - ✓ Una classe di priorità associata al processo a cui il thread appartiene
  - ✓ Una priorità relativa del thread

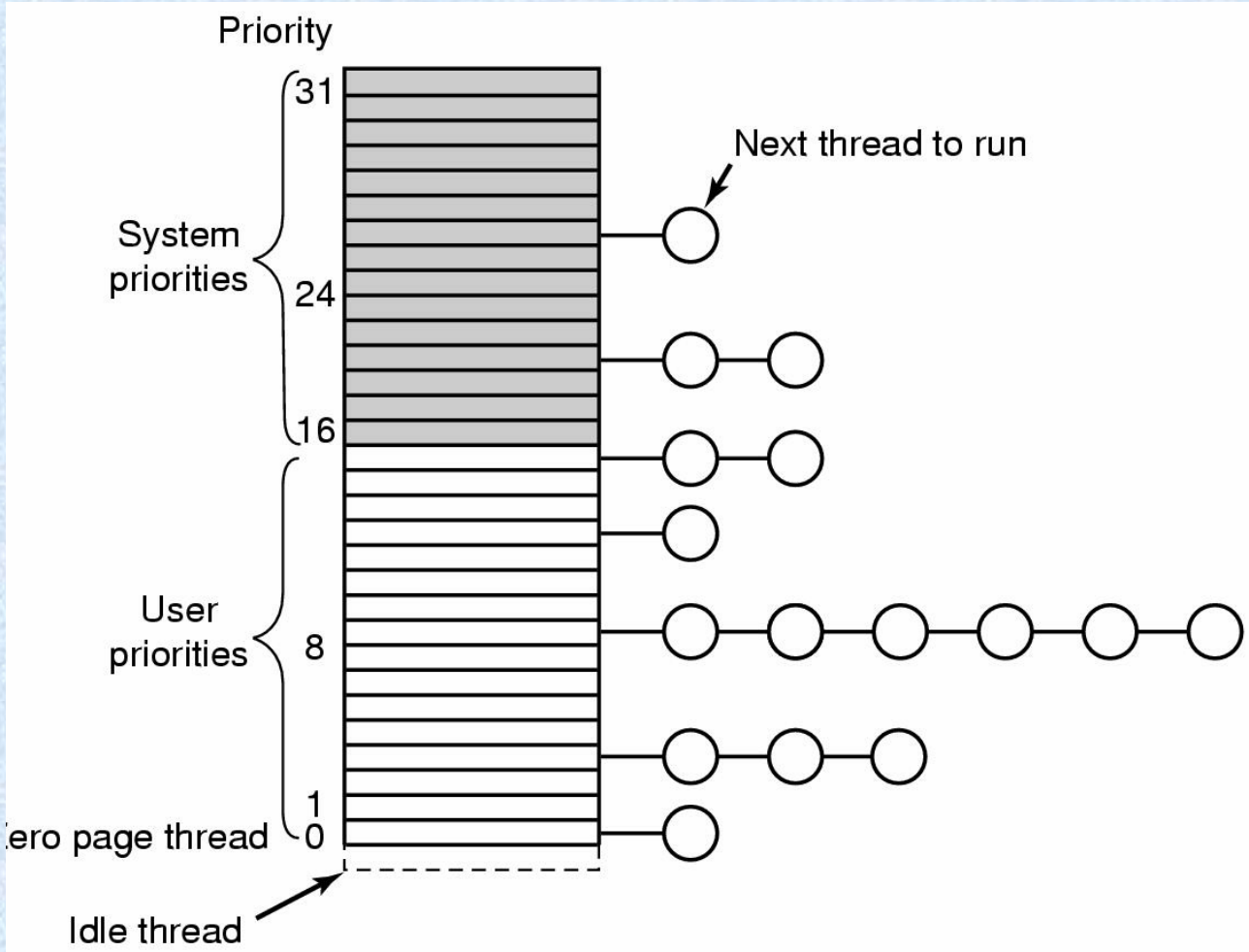
8.2 Scheduling dei thread

- **Classi di priorità di un processo (in ordine decrescente):**
  - ✓ **Real-time**
  - ✓ **High**
  - ✓ **Above Normal**
  - ✓ **Normal**
  - ✓ **Below Normal**
  - ✓ **Idle**

8.2 Scheduling dei thread

- **Priorità relative di un thread (in ordine decrescente):**
  - ✓ **Time Critical**
  - ✓ **Highest**
  - ✓ **Above Normal**
  - ✓ **Normal**
  - ✓ **Below Normal**
  - ✓ **Lowest**
  - ✓ **Idle**

8.2 Scheduling dei thread



8.2 Scheduling dei thread

Algoritmo di scheduling :

Si esegue il primo thread della prima coda non vuota per massimo 1 *quanto* (20ms--120ms)

Scheduling round robin fra thread con la stessa priorità

Come variano le priorità nel tempo :

- **i thread tipicamente entrano a priorità 8**
- **la priorità viene elevata se:**
  - **viene completata una operazione di I/O (+1 disco, +2 linea seriale, +6 tastiera, +8 scheda audio …)**
  - **termina l'attesa su un semaforo, mutex, evento (+1 background, +2 foreground)**
  - **l'input nella finestra di dialogo associata al thread è pronto**

8.2 Scheduling dei thread

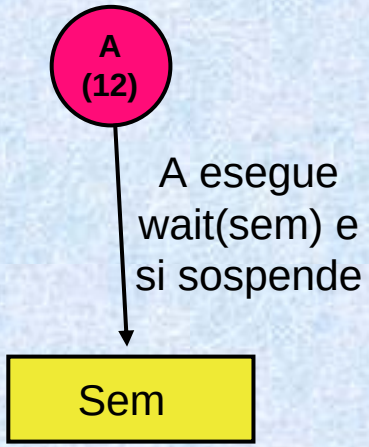
Algoritmo di scheduling :

Come variano le priorità nel tempo (cont.):

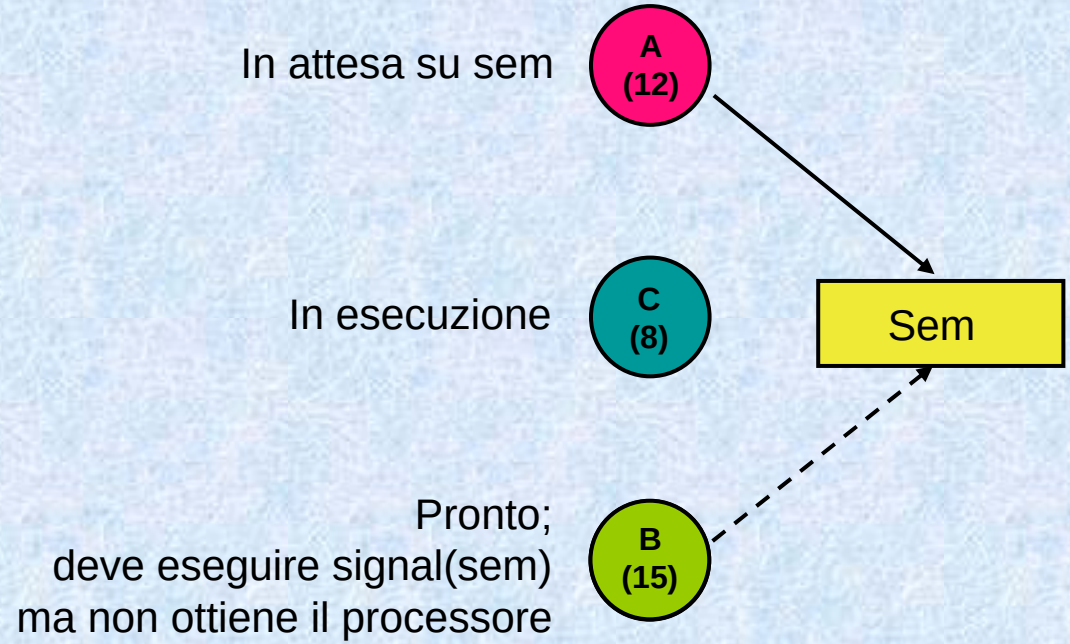
- **la priorità viene abbassata se:**
  - **un thread usa tutto il suo quanto (-1)**
- **se un thread non ha girato per un tempo maggiore di una soglia fissata, allora passa per 2 quanti a priorità 15 (serve a gestire potenziali inversioni di priorità)**

Quando una finestra va in foreground il quanto dei thread corrispondenti viene allungato

8.2 Scheduling dei thread



1)



2)

Un esempio di inversione di priorità

8.3 Sincronizzazione fra thread

- **Esistono due tipi di meccanismi di sincronizzazione:**
  - ✓ **Un insieme di meccanismi "leggeri': *funzioni interlocked* (es. spin-lock)**
  - ✓ **Un insieme di meccanismi per sincronizzazioni più complesse: tramite "oggetti di nucleo" (es. semafori, mutex, ecc.)**

8.3 Sincronizzazione fra thread

Sezioni Critiche

- locali ai thread di un processo
- implementate interamente in spazio utente
- EnterCriticalSection() / LeaveCriticalSection()

Mutex

- ReleaseMutex() corrisponde alla *signal*
- WaitForSingleObject() corrisponde alla *wait*

Eventi

- due stati *set* / *cleared*
- attesa su evento : WaitForSingleObject()
- SetEvent() segnala che l'evento si è verificato

8.3 Sincronizzazione fra thread

Semafori

- creati con CreateSemaphore()
- ReleaseSemaphore() corrisponde alla *signal*
- WaitForSingleObject() corrisponde alla *wait*

Pipe

- byte : funzionano come in Unix
- message : preservano i limiti dei singoli messaggi
- named pipe : possono essere utilizzate anche in rete

Mailslots

- simili ai pipe
- permettono di aver più ricevitori e di inviare messaggi in broadcast