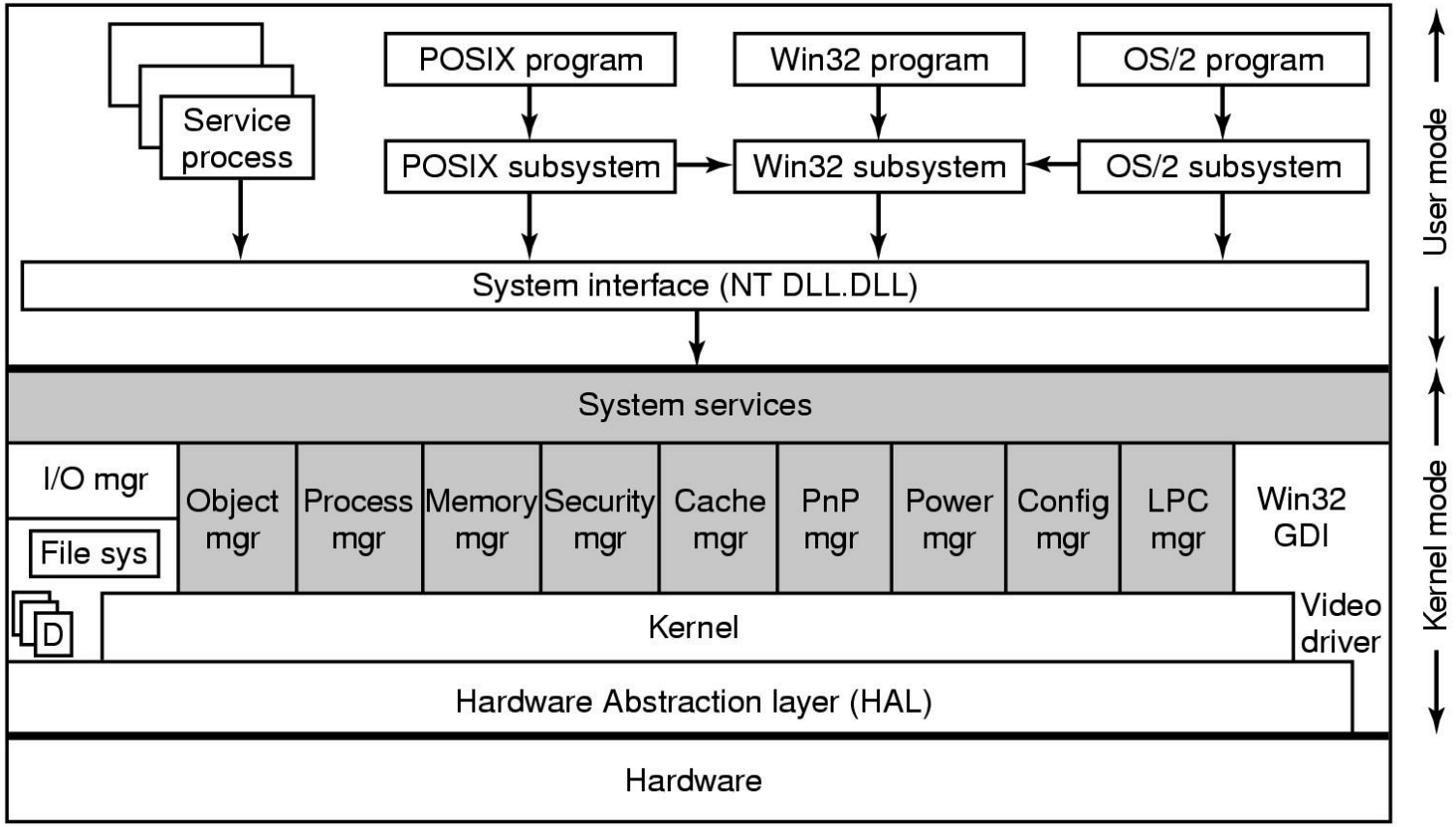
Processi e Thread

- Processi
- Thread
- Meccanismi di comunicazione fra processi (IPC)
- Problemi classici di IPC
- Scheduling
- Processi e thread in Unix
- Processi e thread in Windows

The Operating System Structure



- Structure of Windows 2000 (slightly simplified).
- Shaded area is executed
- Boxes, D, are device drivers
- Service processes are system daemons

Implementation of Objects

Type	Description
Process	User process
Thread	Thread within a process
Semaphore	Counting semaphore used for interprocess synchronization
Mutex	Binary semaphore used to enter a critical region
Event	Synchronization object with persistent state (signaled/not)
Port	Mechanism for interprocess message passing
Timer	Object allowing a thread to sleep for a fixed time interval
Queue	Object used for completion notification on asynchronous I/O
Open file	Object associated with an open file
Access token	Security descriptor for some object
Profile	Data structure used for profiling CPU usage
Section	Structure used for mapping files onto virtual address space
Key	Registry key
Object directory	Directory for grouping objects within the object manager
Symbolic link	Pointer to another object by name
Device	I/O device object
Device driver	Each loaded device driver has its own object

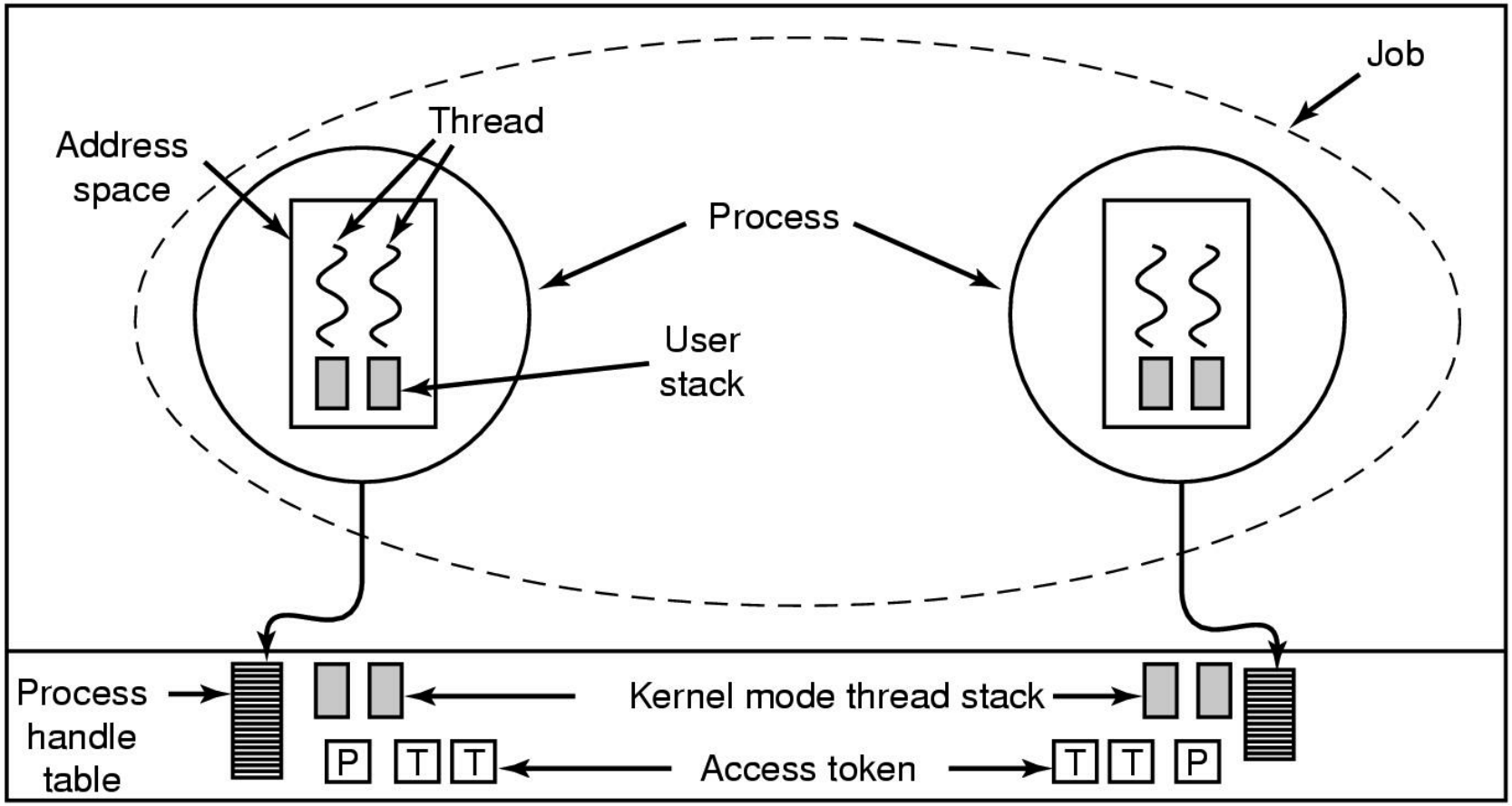
Some common executive object types managed by the object manager

Windows 2000: Processi e Thread (1)

Name	Description
Job	Collection of processes that share quotas and limits
Process	Container for holding resources
Thread	Entity scheduled by the kernel
Fiber	Lightweight thread managed entirely in user space

Unità di esecuzione di base utilizzate per la gestione della CPU e delle risorse

Processi e Thread (2)



Relazione fra job, processi, thread e fibre

Win32 system call per la gestione di Job, Processi, Thread & Fibre

Win32 API Function	Description
CreateProcess	Create a new process
CreateThread	Create a new thread in an existing process
CreateFiber	Create a new fiber
ExitProcess	Terminate current process and all its threads
ExitThread	Terminate this thread
ExitFiber	Terminate this fiber
SetPriorityClass	Set the priority class for a process
SetThreadPriority	Set the priority for one thread
CreateSemaphore	Create a new semapahore
CreateMutex	Create a new mutex
OpenSemaphore	Open an existing semaphore
OpenMutex	Open an existing mutex
WaitForSingleObject	Block on a single semaphore, mutex, etc.
WaitForMultipleObjects	Block on a set of objects whose handles are given
PulseEvent	Set an event to signaled then to nonsignaled
ReleaseMutex	Release a mutex to allow another thread to acquire it
ReleaseSemaphore	Increase the semaphore count by 1
EnterCriticalSection	Acquire the lock on a critical section
LeaveCriticalSection	Release the lock on a critical section

Alcune chiamate di sistema per la gestione delle entità di esecuzione

Thread e Fibre

- Sono implementati nel kernel
- Solo i thread hanno uno stato, lo scheduler lavora solo sui thread
- Ogni thread può ospitare più fibre
- Le fibre sono thread user-level
- Le chiamate Win32 che realizzano le fibre non sono vere system call (sono eseguite interamente in spazio utente)

Windows

Win 32 API

Create Process

- nome eseguibile
- linea di comando
- security descriptors del processo
- security descriptors del thread iniziale
- altre informazioni (es. ambiente)
- directory di lavoro del process
- descrizione windows
- struttura per parametri di ritorno (es. gestore nouvo processo)

Windows

Win 32 API

Create Thread

- security descriptor
- dimensione iniziale stack
- starting address
- stato iniziale (pronto, bloccato)
- TID

Windows: Meccanismi di IPC (1)

- Pipe (bidirezionali)
  - byte : funzionano come in Unix
  - message : preservano i limiti dei singoli messaggi
  - named pipe (con nome): possono essere utilizzate anche in rete
- Mailslots
  - simili ai pipe (ma non bidirezionali)
  - permettono di aver più ricevitori e di inviare messaggi in broadcast
- Semafori
  - creati con CreateSemaphore()
  - ReleaseSemaphore() corrisponde alla *up*
  - WaitForSingleObject() corrisponde alla *down*
- Socket (reti)
- Chiamate di procedure remote
- Memoria condivisa

Windows Meccanismi di IPC (2)

- Mutex
  - ReleaseMutex() corrisponde alla *up*
  - WaitForSingleObject() corrisponde alla *down*
- Sezioni Critiche
  - locali al thread che le ha create
  - implementate interamente in spazio utente
  - EnterCriticalSection() / LeaveCriticalSection()
- Eventi
  - sue stati *set* / *cleared*
  - attesa su evento : WaitForSingleObject()
  - SetEvent() segnala che l'evento si è verificato

Scheduling in Windows 2000 (1)

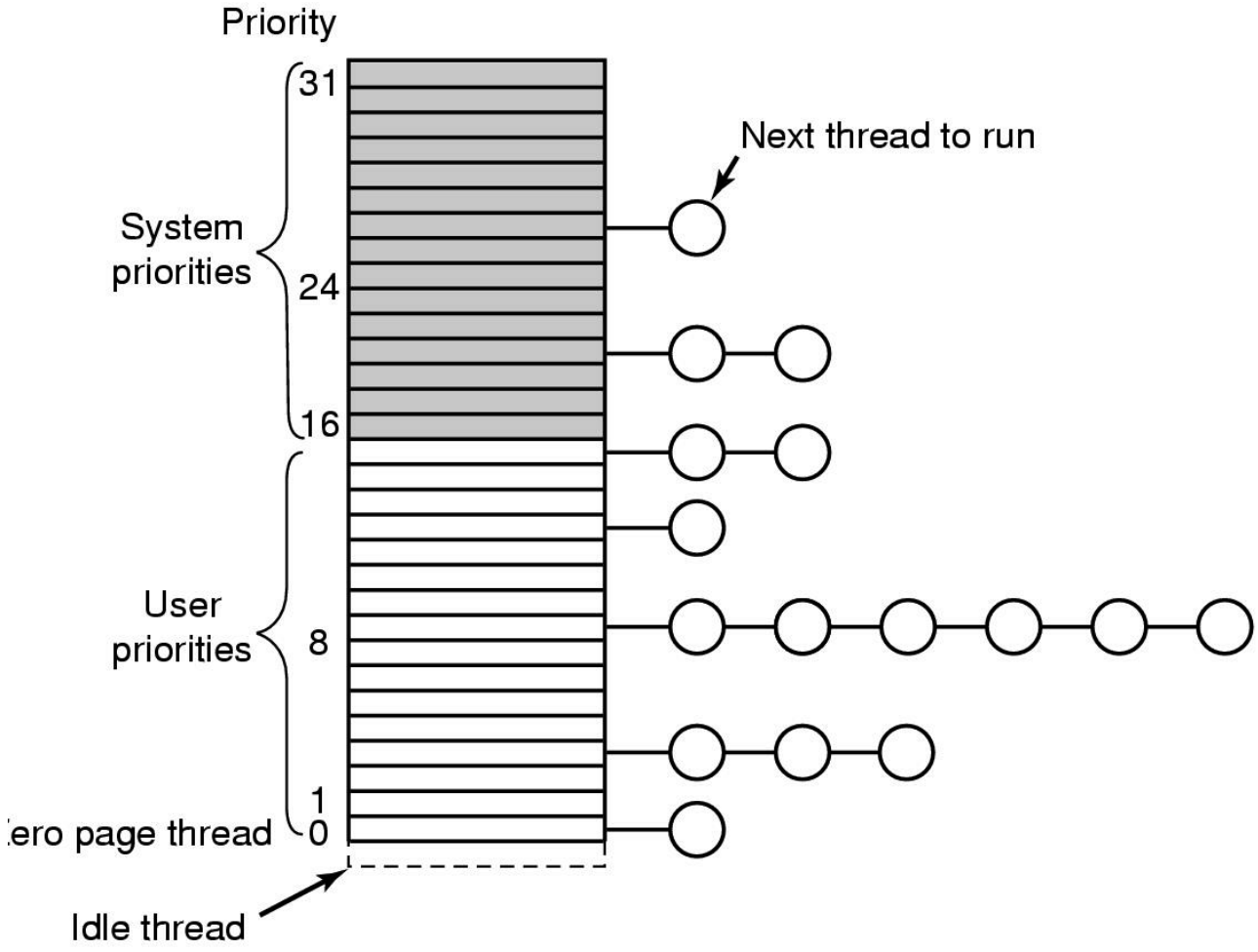
- Win32 permette all'utente di specificare :
  - priorità di un processo (6 livelli diversi)
  - priorità di un thread all'interno di un processo (7 livelli diversi)
- Windows 2000 mappa le 42 combinazioni possibili su 32 livelli di priorità

Scheduling in Windows 2000 (2)

		Win32 process class priorities					
		Realtime	**High**	**Above Normal**	**Normal**	**Below Normal**	**Idle**
Win32 thread priorities	**Time critical**	31	15	15	15	15	15
	Highest	26	15	12	10	8	6
	Above normal	25	14	11	9	7	5
	Normal	24	13	10	8	6	4
	Below normal	23	12	9	7	5	3
	Lowest	22	11	8	6	4	2
	Idle	16	1	1	1	1	1

Corrispondenza fra le priorità di Win32 e quelle di Windows 2000

Scheduling in Windows 2000 (3)



Windows 2000 fornisce 32 priorità diverse per i thread

Scheduling in Windows 2000 (4)

Algoritmo di scheduling :

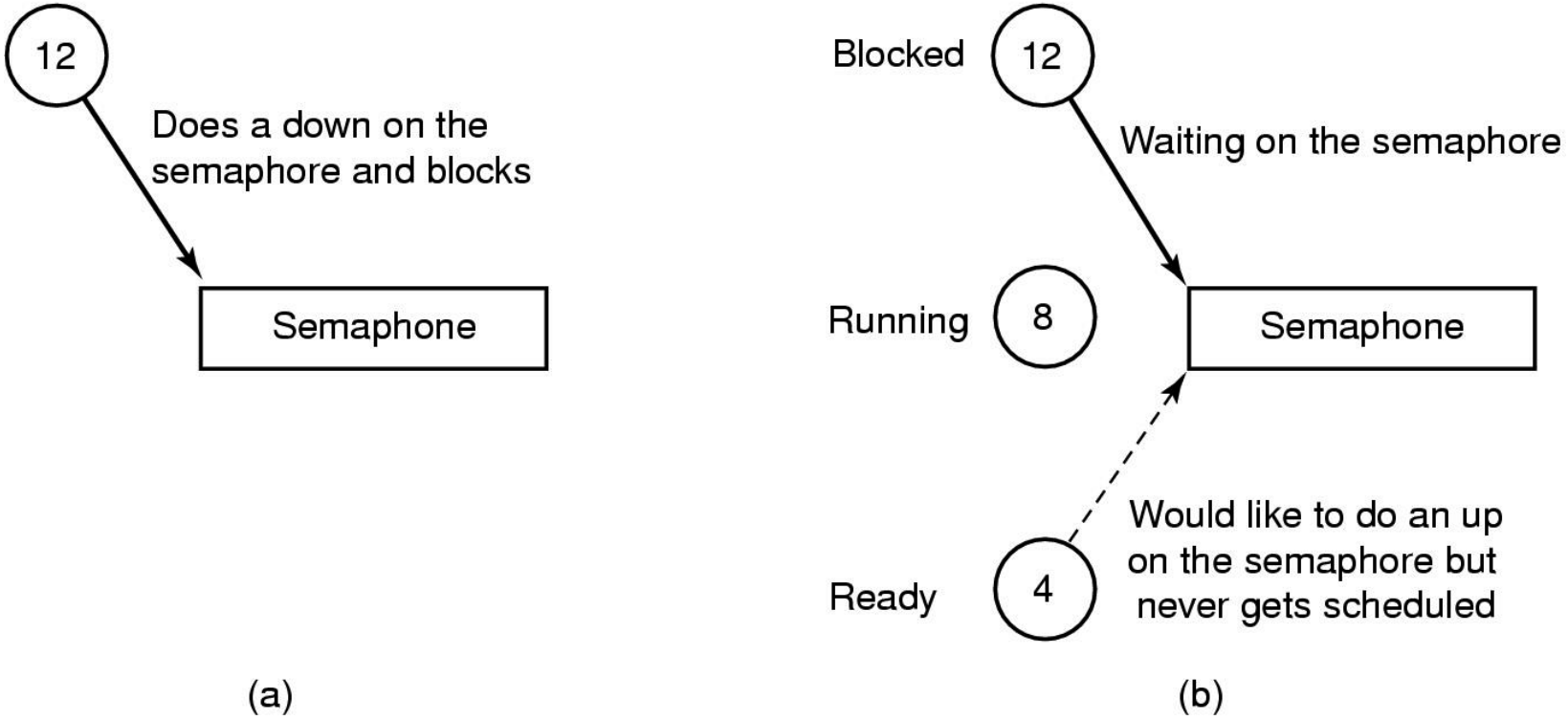
- Si esegue il primo thread della prima coda non vuota per massimo 1 *quanto* (20ms--120ms)
- Scheduling round robin fra thread con la stessa priorità
- Come variano le priorità nel tempo :
  - i processi tipicamente entrano a priorità 8
  - la priorità viene elevata se:
    - viene completata una operazione 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

Scheduling in Windows 2000 (5)

Algoritmo di scheduling :

- Come variano le priorità nel tempo (cont.):
  - la priorità viene abbassata se:
    - un thread usa tutto il suo quanto (-1)
    - un thread non ha girato per un tempo maggiore di una soglia fissata (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

Scheduling in Windows 2000 (6)



Un esempio di inversione di priorità