

Sistemi Operativi

Modulo 11: Windows 2000

Renzo Davoli
Alberto Montresor

Copyright © 2002-2005 Renzo Davoli, Alberto Montresor
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license can be found at: <http://www.gnu.org/licenses/fdl.html#TOC1>

Windows NT-2000-XP

• **Caratteristiche**

- s.o. a 32 bit, con scheduling preemptive
- è portabile
 - Windows NT: 386+, R4000, DEC Alpha, Motorola PowerPC
 - Windows 2000/XP: Pentium / IA64
- è compatibile
 - all'indietro: MS-DOS, Windows 9x
 - con lo standard POSIX
- è estendibile
- fornisce supporto per
 - sistemi multiprocessore
 - sicurezza
 - affidabilità

© 2002-2005 Renzo Davoli, Alberto Montresor

2

Storia

• **Primi anni '80**

- Microsoft e IBM cooperarono per sviluppare OS/2
- OS-2 era scritto in assembler per 80286 ed forniva supporto per macchine uniprocessore
- Windows NT parte da questa esperienza

• **Ottobre 1988**

- parte il progetto Windows NT con lo scopo di supportare le API sia POSIX sia OS-2
- durante il progetto viene aggiunta anche Win32.

© 2002-2005 Renzo Davoli, Alberto Montresor

3

Storia

• **1993**

- esce la prima versione di NT (3.1 e 3.1 Advanced Server)
- il nome è dovuto a ragioni di marketing; Windows 16 bit era allora alla versione 3.1

• **1996**

- esce Windows NT 4.0
- viene adottato lo standard grafico di Windows 9x
- alcune routine dell'interfaccia grafica vengono spostate nel kernel
 - aumentano le prestazioni, ma diminuisce l'affidabilità

© 2002-2005 Renzo Davoli, Alberto Montresor

4

Storia

- **1999**
 - esce Windows 2000
 - non è altro che Windows NT 5.0 (vedi version number di \winnt\system32\ntoskrnl.exe)
 - aggiunge
 - supporto per architetture plug-and-play
 - support per USB, FireWire, IrDA, power management
 - active directory service
 - internazionalizzazione (i18n)
- **2001**
 - esce Windows XP
 - nuova interfaccia grafica...

Principi di progettazione

- **Estensibilità**
 - basato su *environmental subsystem* per emulare diversi ambienti operativi (Win32, MS-DOS, Win16, OS/2, POSIX)
- **Portabilità:**
 - è scritto in C e C++ quindi portabile
 - l'interfacciamento all'hardware è effettuato tramite HAL.DLL (hardware abstraction layer, dynamic link library)

Principi di progettazione

- **Compatibilità**
 - può eseguire codice nativo per diverse architetture (MS-DOS, Win16, Win32, OS-2, LanManager)
 - anche su architetture con processore diverso, con emulazione software
 - fornisce compatibilità source-code con POSIX
 - le system call/chiamate al BIOS vengono emulate dall'*environmental subsystem* corrispondente
 - vengono gestiti diversi tipi di File System (FAT, NTFS, HPFS, ISO9660)
 - La compatibilità non è perfetta: l'accesso diretto all'hardware (e.g. porte I/O) è vietato.

Principi di progettazione

- **Affidabilità**
 - utilizza strutture di protezione alla memoria e al file system.
 - il file system NTFS ha meccanismi di recovery per vari tipi di errore e system crash
 - NT viene classificato C-2 (moderate level of protection) dal governo americano
- **Performance**
 - supporta multiprocessore (con un grado di scalabilità minore di UNIX)
 - meccanismi efficienti per comunicazione interna
 - scheduler a priorità

API Win32

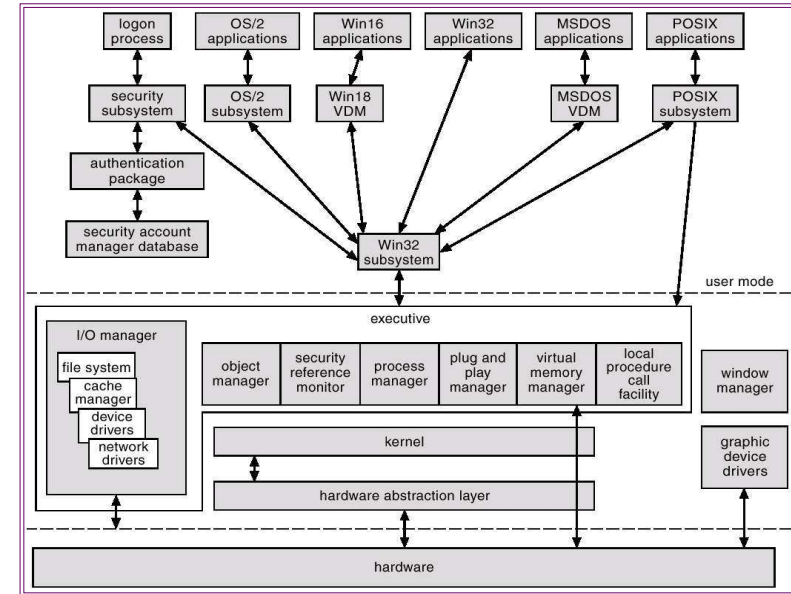
• System call

- Windows 2000 è dotato di un insieme di system call per il kernel
- Microsoft non ha mai pubblicato una lista delle system call comprese in questo insieme
- le system call variano da versione a versione

• API Win 32

- è un insieme di funzioni pubbliche e documentate
- funzioni di libreria che svolgono il loro compito in user mode oppure invocano le opportune system call
- comprende letteralmente migliaia di funzioni

Windows 2000 - Architettura



Windows 2000 - HAL

• Hardware Abstraction Layer (HAL)

- strato software con lo scopo di nascondere al resto del s.o. le peculiarità dell'hardware sottostante
- fornisce un'interfaccia comune per un insieme di servizi low-level
 - gestione degli interrupt
 - gestione del timer
 - gestione del DMA
 - device I/O
 - spin lock
 - comunicazione con il BIOS
- serve a migliorare la portabilità del s.o. (idealmente)
- dipende dalla versione hardware della macchina

Windows 2000 - Kernel

• Il "kernel" di Windows 2000

- secondo Microsoft, lo strato di software compreso tra HAL e l'Executive è detto "kernel" o addirittura "microkernel"
- in realtà, a partire da Windows NT 4.0, la totalità del s.o. (sottosistema grafico incluso), vive ormai in kernel mode

• Responsabilità del Kernel

- scheduling dei thread
- gestione degli interrupt e delle eccezioni
- meccanismi di sincronizzazione

Windows 2000 - Kernel

- **Caratteristiche**
 - è ancora parzialmente "platform-dependent"
 - utilizza i servizi di HAL per costruire servizi di più alto livello
- **Esempio:**
 - HAL fornisce meccanismi per associare un interrupt handler ad un interrupt
 - il Kernel fornisce un meccanismo per gestire i context switch (ovvero salvataggio/ripristino registri) per gli interrupt handler
- **Il kernel è object-oriented; due tipi principali di oggetti:**
 - control object
 - dispatcher object

Windows 2000 - Kernel

- **Control object**
 - contengono informazioni necessarie per il controllo del sistema
- **Esempi**
 - *Interrupt Object*
 - lega un'interrupt handler ad una sorgente di interrupt
 - *Process Object*
 - process control block per Windows 2000
 - *Asynchronous/Deferred Procedure Call (APC/DPC)*
 - interrompono un thread ed eseguono una procedura
 - vedi lucidi seguenti
 - *Profile Object*
 - misura il tempo impiegato da un certo blocco di codice

Windows 2000 - Il Kernel

- **Dispatcher object**
 - ovvero, sono gli oggetti per la sincronizzazione
- **Esempi**
 - *Eventi*
 - associazione evento-gestore, registrazione dell'occorrenza
 - *Mutanti*
 - per mutua esclusione
 - *Mutex*
 - mutua esclusione per processi in kernel mode; deadlock free
 - *Semaphore*
 - *Thread*
 - *Timer*

Windows 2000 - Scheduling

- **Quattro modalità diverse di scheduling**
 - Job
 - Processi
 - Thread
 - Fiber
- **Job**
 - è una collezione di uno o più processi che vengono gestiti come una singola unità
 - ogni job è associato a:
 - ad un insieme di limiti di risorse e quote
 - ad es., il numero massimo di processi nel job
 - la massima quantità di CPU da assegnare ai processi
 - la massima quantità di memoria da assegnare ai processi

Windows 2000 - Scheduling

• Processi

- i processi sono contenitori per risorse
- ogni processo è associato a:
 - un process ID
 - uno spazio di indirizzamento virtuale, fino a 2-3GB
 - una lista di handle per le diverse risorse esistenti
 - un access token utilizzato per accedere alle risorse
- ogni processo è formato da uno o più thread

Windows 2000 - Scheduling

• Thread

- sono alla base dello scheduling della CPU
 - lo scheduler associa uno stato di esecuzione ai thread e non ai processi
 - lo scheduler seleziona il prossimo thread da eseguire e non il prossimo processo
- i thread si dividono in
 - *user thread*
 - associati allo user process corrispondente
 - *kernel thread*
 - sono associati ad uno speciale processo di sistema
 - eseguono compiti che possono essere eseguiti in background, ad esempio gestione della memoria

Windows 2000 - Scheduling

• Fiber

- fare context switch tra thread è costoso perchè richiede il passaggio in kernel mode
- fiber sono "lightweight" thread, in quanto vengono schedulati a livello utente

• Implementazione

- il kernel di Windows 2000 non sa nulla relativamente ai fiber
- lo scheduling dei fiber è gestito da una libreria a livello utente a livello Win 32

Windows 2000 - Scheduling

• Process API - più di cento chiamate diverse

- CreateProcess
- CreateThread
- CreateFiber
- ExitProcess
- ExitThread
- ExitFiber
- SetPriorityClass
- SetThreadPriority
- CreateSemaphore
- CreateMutex
- OpenSemaphore
- OpenMutex
- WaitForSingleObject
- WaitForMultipleObject
- ReleaseMutex
- ReleaseSemaphore
- EnterCriticalSection
- ExitCriticalSection

Windows 2000 - Scheduling

- **Stati di un thread in Windows 2000**
 - *ready*
 - *running*
(uno per ogni processore)
 - *stand-by*
(il thread a priorità più alta in attesa, il prossimo, uno per processore)
 - *waiting*
(e.g. pending I/O)
 - *transition*
(non è ancora partito perché in attesa di qualche risorsa indisponibile)
 - *terminated*

Windows 2000 - Scheduling

- **Dal punto di vista delle API Win32**
 - ogni processo ha una *classe di priorità*
(realtime, high, above normal, normal, below normal, idle)
 - ogni thread ha una *priorità*
(time critical, highest, above normal, normal, below normal, slowest, idle)
- **Dal punto di vista dello scheduler nel Kernel**
 - esistono 32 livelli di priorità
- **Esiste una tabella che mappa i $6*7=42$ stati delle API nei 32 stati del Kernel (!)**

Windows 2000 - Scheduling

- **I 32 livelli di priorità sono così suddivisi:**
 - 0-15 per thread utenti
 - 16-31 per thread "di sistema" (detti anche real-time)
- **Nota:**
 - Windows 2000 NON E' un sistema operativo real-time
- **Ogni thread ha due valori di priorità:**
 - base priority (statica, assegnata tramite API Win32)
 - current priority (dinamica, dipende dall'esecuzione del thread)

Windows 2000 - Scheduling

- **Scheduler**
 - preemptive e time-sharing
 - composto da 32 code di processi ready, una per livello di priorità
 - ad ogni istante, viene selezionato un processo dalla coda non-vuota con priorità più alta

Windows 2000 - Scheduling

- **Nel caso un thread termini il proprio time slice:**
 - la sua priorità viene abbassata di 1
 - mai sotto alla sua base priority
- **Nel caso un thread venga risvegliato:**
 - la sua priorità viene incrementata di una quantità che dipende dal motivo del risveglio (dispositivo, meccanismo di sincronizzazione)
 - 1 disco, 2 linea seriale, 6 tastiera/mouse, 8 sound card
 - 1-2 in caso di risveglio da semaforo o mutex
- **Nota:**
 - esistono anche altri "hack" abbastanza strani al fine di evitare fenomeni di starvation...

Windows 2000 - Gestione delle eccezioni

- **In kernel mode:**
 - vengono gestite dall'exception dispatcher del kernel che cerca un gestore (handler) opportuno
 - se non viene trovato viene originato un errore fatale e si genera il "blue screen of death"
- **In user mode**
 - ogni sottosistema può avere un proprio debugger: quindi
 - prima si cerca il debugger (se gestisce l'eccezione)
 - altrimenti si cerca un handler specifico
 - altrimenti si manda come messaggio al sottosistema specifico (e.g. per trasformarlo in segnale POSIX)
 - altrimenti si termina il processo.

Windows 2000 - Gestione delle eccezioni

- **HAL trasforma gli interrupt (dipendenti dall'architettura) in un insieme standard:**
 - Livello Tipo
 - 31 hardware failure o bus error
 - 30 power failure
 - 29 inter-processor notification
 - 28 clock
 - 12-27 IRQ hardware
 - 4-11 kernel (dispatch & deferred proc.call)
 - 3 software debugger
 - 0-2 software interrupts (device drivers)

Windows 2000 - Gestione delle eccezioni

- **Ogni processore indipendentemente stabilisce il proprio livello di interrupt.**
- **Se il livello di un processore è N**
 - tutti gli interrupt di livello $\leq N$ sono mascherati
 - gli altri sono attivi
- **Interrupt & SMP**
 - Windows 2000 usa lock in memoria per sincronizzare l'accesso a strutture dati condivise da parte di thread in esecuzione su processori distinti
 - siccome un processore si ferma quando tenta di acquisire il lock, un thread che detiene un lock viene eseguito in modo non interrompibile così da rilasciare il lock ASAP.

Windows 2000 - APC e DPC

- **Deferred Procedure Calls**
 - è utilizzato per separare la gestione "immediata" di un interrupt dalla gestione vera e propria
 - esempio: pressione di un tasto
 - la routine di gestione dell'interrupt registra il tasto premuto
 - invoca una deferred procedure call
 - successivamente, la deferred procedure call viene eseguita
 - il meccanismo delle DPC permette di ricordare che c'è ancora lavoro da fare...
- **Asynchronous Procedure Call**
 - come DPC, ma per processi utenti

Windows 2000 - Executive

- **E' il componente che fornisce servizi agli environmental subsystem**
 - object manager
 - virtual memory manager
 - process manager
 - local procedure call facility
 - I/O manager / cache manager
 - security reference monitor
 - power manager
 - configuration manager

Windows 2000 - Executive

- **Object Manager**
 - in un s.o. object-oriented, ogni entità (thread, processi, file, directory, etc.) sono rappresentati da oggetti
 - l'object manager è responsabile per la gestione
 - quando un thread vuole accedere ad un oggetto lo apre con la funzione *open*
 - è il luogo naturale dove porre i controlli di sicurezza: all'atto della open si controlla se il thread può o meno accedere all'oggetto
 - la *open* ritorna un "handle" all'oggetto

Windows 2000 - Executive

- **Oggetti temporanei e oggetti permanenti**
 - oggetti permanenti:
 - rappresentano entità fisiche del sistema, i.e. dischi.
 - oggetti temporanei: rappresentano entità logiche che hanno durata limitata: thread, processi...
- **L'object manager**
 - tiene sempre il numero dei riferimenti attivi per ogni oggetto
 - gli oggetti temporanei vengono eliminati quando questo contatore diventa zero

Windows 2000 - Executive

- **Operazioni sugli oggetti**
 - Open
 - Close
 - Create
 - Delete
 - Query Name: handle -> nome
 - Parse: Nome-> handle
 - Security: per cambiare i diritti di un oggetto.

Windows 2000- Executive

- **Object name**
 - gli oggetti possono essere associati a nomi , in modo che processi differenti possano accedervi
- **Organizzazione**
 - hanno una struttura gerarchica (come il file system)
 - esistono i symbolic link object
 - lo spazio di nomi può venir ampliato aggiungendo degli object domain (e.g. quando si inserisce un floppy)
 - hanno una Access Control List associata
 - c'è un campo di audit (per tenere traccia delle operazioni effettuate)

Windows 2000- Executive

- **Virtual-Memory Manager**
 - Pagine da 4 KB
 - le pagine assegnate a un processo ma non in memoria centrale sono memorizzate in uno o più paging file
 - l'indirizzamento è a 32 bit (4GB indirizzabili)
 - i 2GB "alti" sono accessibili solo in kernel mode e sono comuni a tutti i processi
 - i 2GB "bassi" sono accessibili anche in user mode e sono privati di ogni processo

Windows 2000- Executive

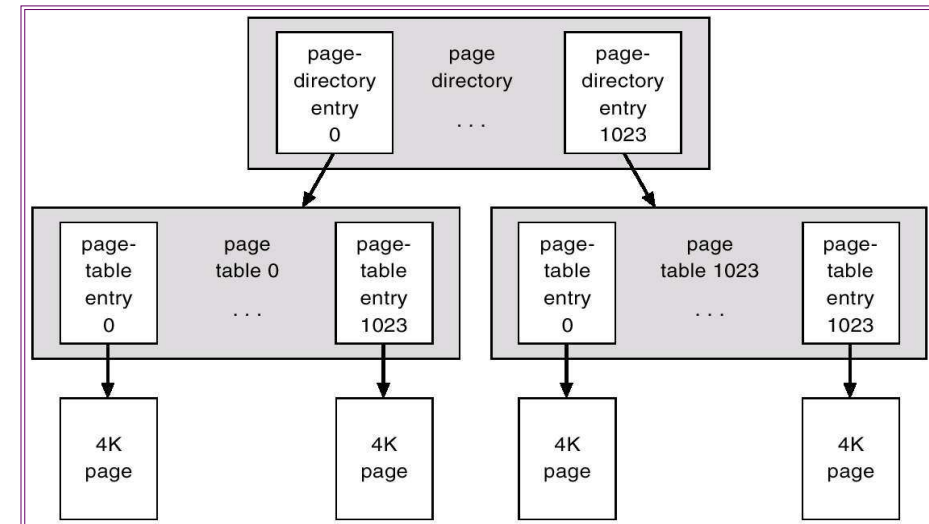
- **Virtual-Memory Manager**
 - anche le aree di memoria sono considerate oggetti da NT
 - l'allocazione avviene in due fasi:
 - reservation: viene richiesto un certo spazio di indirizzamento
 - commitment: viene allocata una certa area di paging file
 - l'object manager può controllare limiti di allocazione (quota)
 - in Windows 2000 il processo "padre" può accedere alla memoria dei processi che ha generato (e.g. serve agli environmental subsystem).

Windows 2000- Executive

- **Due processi possono chiedere di accedere allo stesso oggetto di memoria**
- **Lo condividono ma:**
 1. risulta come spazio utilizzato da ogni processo (se molto ampio si possono superare i limiti)
 2. può essere posto ad un indirizzo diverso per ogni processo
- **Per risolvere (1) si possono usare i section object:**
 - ogni processo ha una view sull'oggetto (come una finestra) e "vede" solo una porzione dell'oggetto
 - porzione che può far "camminare" sull'oggetto
- **Per risolvere (2) si può usare memoria "based":**
 - viene mappata allo stesso indirizzo in tutti i processi

Windows 2000- Executive

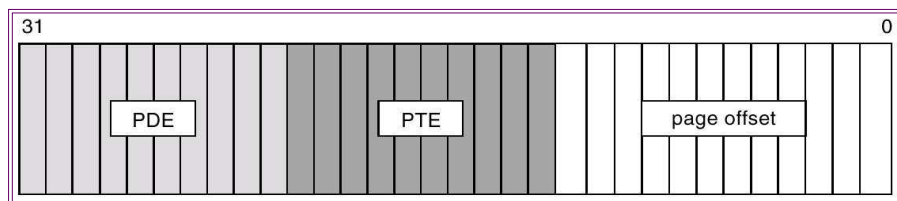
- **Memory Manager: strutture dati**



Windows 2000- Executive

- **Struttura di un indirizzo virtuale**

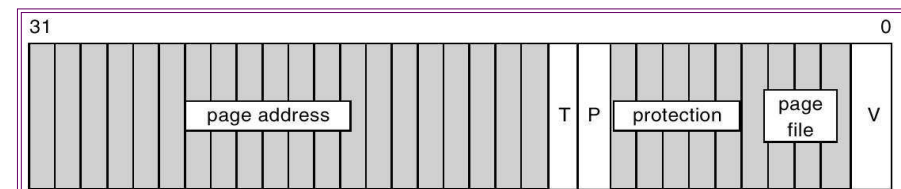
- PDE: page directory entry
- PTE: page table entry



Windows 2000- Executive

- **Struttura di una PTE**

- valid (usata)
- free (libera non referenziata)
- zeroed (ripulita, pronta per essere riciclata)
- standby (temporaneamente rimossa al processo)
- modificata (da salvare sul paging file)
- bad (inusabile a causa di un errore hardware)



Windows 2000- Executive

• Memory manager

- per le pagine condivise invece che avere due elementi di PTE che puntano allo stesso frame si fa puntare gli elementi di PTE a un prototype PTE (che punta al frame) così non si duplicano aggiornamenti
- c'è una funzionalità di prefetching delle pagine adiacenti a quella utilizzata
- a tutti i processi viene inizialmente allocato uno spazio di 30 pagine in memoria centrale (impropriamente chiamato working set)
- per vedere se si può ridurre, NT "ruba" una pagina al processo e verifica se avviene un page fault

Windows 2000 - Executive

• Process manager

- responsabile per la creazione di processi e thread
- non supporta nessun tipo di relazione tra processi (tipo parentela e/o gerarchia)
- lascia che queste informazioni aggiuntive vengano trattate dagli environmental subsystem

Windows 2000 - Executive

• Local Procedure Call

- implementa una interazione client-server su singola macchina
- utilizzata dagli E.S. per richiedere servizi all'Executive
- viene creato un canale bidirezionale di comunicazione fra client e server (per passare i parametri prima e il risultato al termine)

• Esistono tre tecniche per passare i parametri:

- per piccoli messaggi: si usa una coda standard (quella creata automaticamente)
- per messaggi più grandi
 - tramite memoria condivisa
 - nel canale viene passato il riferimento alla memoria condivisa
- quick LPC

Windows 2000 - Executive

• Quick LPC

- usate da Win32 per dialogare con i dispositivi grafici

• Coinvolge tre oggetti

- un server thread (uno per ogni client thread)
 - per eliminare l'overhead per identificare il cliente
 - hanno la preferenza dello scheduler
- una sezione di memoria condivisa
 - per eliminare l'overhead della copia
- un event-pair object
 - fornisce segnalazioni dell'avvenuto scambio di informazioni fra il client thread e Win 32
 - consente di evitare l'uso del canale di comunicazione

Windows 2000 - Executive

I/O Manager

- è responsabile della gestione dei driver
 - inclusi i "driver" per i vari file system supportati
 - inclusi i "driver" per la rete
- le richieste all'I/O manager hanno un formato standard chiamato *I/O request packet* (IRP)
- l'I/O manager funge da dispatcher degli IRP (quando riceve una richiesta la indirizza al driver opportuno)
- l'I/O manager implementa il meccanismo di plug-and-play

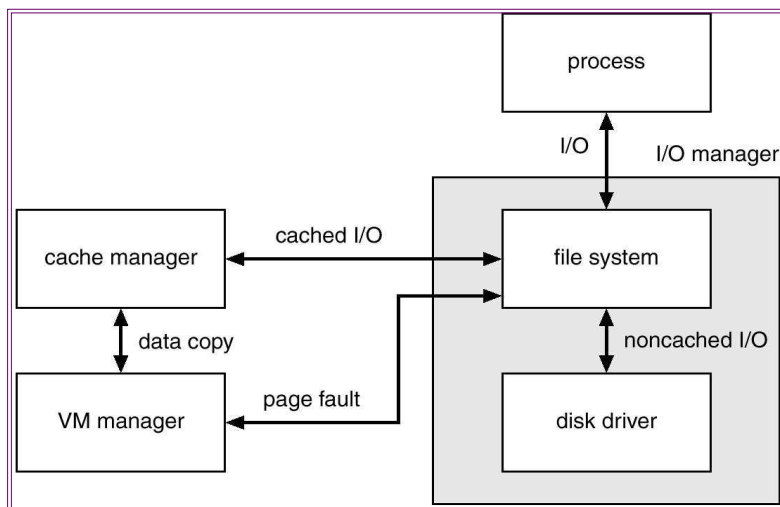
Windows 2000 - Executive

Cache Manager

- Windows 2000 utilizza un'architettura di caching centralizzata
- usa 1GB nella metà alta dello spazio di indirizzamento virtuale (kernel mode comune a tutti i processi)
- concetti fondamentali
 - i file vengono mappati in memoria virtuale
 - il memory manager viene utilizzato per gestire i page fault nel caso le pagine accedute non siano in memoria
 - solo il memory manager conosce la quantità di memoria utilizzata per la cache; in questo modo, il memory manager può gestire il trade off fra quantità di memoria destinata ai processi e quantità di memoria destinata alla cache

Windows 2000 - Executive

Cache Manager



Environmental Subsystems

Ogni applicazione

- “vive” all’interno di un sottosistema d’ambiente (*Environmental Subsystem*, ES)
- **Se l’applicazione è Win32**
 - l’ES per Win32 è sempre attivo
- **Se l’applicazione non è Win32**
 - viene controllato se l’ES corrispondente è attivo, se non lo è viene attivato e quindi viene eseguita l’applicazione
- **I sottosistemi sono “mutualmente esclusivi”**: una applicazione Win32 non può chiamare una routine POSIX

MS-DOS Subsystem

- Viene fornita una applicazione chiamata **Virtual Dos Machine** (in breve VDM)
- **La VDM**
 - ha una instruction-execution unit per eseguire (o emulare) le istruzioni x86
 - emula le routine del BIOS e i servizi dell'interrupt 21
- **Applicazioni che accedono direttamente all'hardware o al file system non possono funzionare correttamente**

WIN16 Subsystem

- **E' detto anche Windows on Windows.**
- **Fornisce**
 - l'emulazione delle routine del kernel di Windows 3.1
 - l'emulazione delle routine del Window manager
 - una conversione automatica degli indirizzi da 16 bit a 32 bit
- **Tutte le applicazioni Win16**
 - condividono lo stesso spazio indirizzabile e la stessa coda di input, una sola è attiva alla volta
- **Se una applicazione si blocca o va in crash**
 - allora blocca o può rovinare tutte le altre in esecuzione nell'ambiente Win16

POSIX Subsystem

- **E' composto da tre parti:**
 - posix subsystem server (PSXSS.EXE)
 - dynamic linking library (PSXDLL.DLL)
 - console session manager (POSIX.EXE)
- **Le applicazioni possono accedere al FS**
 - viene forzata la compatibilità con i permessi UNIX-like
- **Molte funzionalità di Win32 risultano però inaccessibili**
 - networking, grafica, etc...

OS/2 Subsystem

- **Fornisce solo funzionalità limitate.**
- **Esiste solo nelle versioni di NT per x86**
- **Le applicazioni a caratteri**
 - possono funzionare nel sottosistema OS/2 (quindi solo su macchine basate su Intel).
- **Le applicazioni dette "Real Mode" di OS/2**
 - funzionano nel sottosistema MS/DOS (quindi anche su altri sistemi)

Logon e Security Subsystem

- **Il sottosistema di autenticazione**
 - dato il login e la password controlla l'identità dell'utente o da un database locale o tramite un server in rete
 - fornisce un *access token*
 - viene utilizzato per ogni richiesta di accesso agli oggetti
 - contiene: i privilegi, i gruppi di appartenenza e i limiti di accesso alle risorse

NTFS

- **Windows 2000**
 - supporta diversi tipi di file system (FAT-16, FAT-32, NTFS, ISO-9660)
 - **Caratteristiche di NTFS**
 - è un file system moderno, di tipo journaling, progettato da zero e non basato sul file system di MS-DOS
 - supporta
 - nomi fino a 255 caratteri, case-sensitive *
 - pathname fino a 32768 caratteri
 - nomi basati su Unicode
 - i file possono raggiungere una lunghezza di 2^{64} byte
- * ma le API Win32 non supportano nomi case-sensitive per le directory

NTFS - Struttura dei file

- **I file NTFS**
 - non sono semplici sequenze lineari di byte
 - sono costituiti da un insieme di *attributi*, ognuno dei quali è composto da
 - una sequenza lineare di byte
 - un nome che permette di identificare l'attributo
- **Normalmente, un file è composto:**
 - da un attributo a lunghezza variabile contenente il nome
 - da uno o più attributi contenenti informazioni sui file (attributi generici di POSIX)
 - da un attributo senza nome contenente i dati

NTFS - Struttura dei file

- **File più complessi**
 - potrebbero essere composti da due o più attributi dati
 - ogni attributo ha una lunghezza massima teorica di 2^{64} byte
 - gli attributi possono essere letti in modo indipendente
 - esempio: file immagine che contengono la stessa immagine a diverse risoluzioni e/o formati
- **Considerazioni**
 - viene estesa l'idea Macintosh (due attributi, data e resource) ad un numero indefinito di "fork"
 - dati e informazioni sui file vengono gestiti in un unico modo

NTFS - API

- CreateFile `open`
- DeleteFile `unlink`
- CloseHandle `close`
- ReadFile `read`
- WriteFile `write`
- SetFilePointer `lseek`
- GetFileAttributes `stat`
- LockFile `fcntl`
- UnlockFile `fcntl`
- CreateDirectory `mkdir`
- RemoveDirectory `rmdir`
- FindFirstFile `opendir`
- FindNextFile `closedir`
- MoveFile `rename`
- SetCurrentDir `chdir`
- Più innumerevoli altre; ad esempio esistono chiamate per copiare file

NTFS - Organizzazione

- **L'entità fondamentale è il volume**
 - il volume può essere contenuto in una partizione, un intero disco o può essere contenuto in più dischi
 - tutti i metadati (informazioni sul volume) sono contenuti in file ordinari all'interno del volume stesso
- **Ogni volume**
 - è visto come una sequenza lineare di blocchi (*cluster*)
 - 1 settore per dischi fino a 512M
 - 1KB per dischi fino a 1GB
 - 2KB per dischi fino a 2GB
 - 4KB per dischi di dimensioni maggiori
 - ogni cluster è indirizzato dal suo *Logical Cluster Number* (LCN)

NTFS: MFT

- **Master File Table (MFT)**
 - in ogni volume è presente una struttura dati chiamata MFT
 - la MFT è una sequenza lineare di record di dimensione 1 KB
 - ogni record descrive un file o una directory (corrisponde più o meno ad un inode)
- **Ogni MFT record contiene:**
 - uno header di record
 - una sequenza di coppie (header di attributo, valore)
 - l'header di attributo contiene un identificatore di attributo e la lunghezza del valore (per attributi a lunghezza variabile)

NTFS: MFT

- **Se l'attributo è piccolo**
 - viene mantenuto all'interno del record e viene detto *resident*
- **Se l'attributo è grande**
 - viene mantenuto esternamente e viene detto *non-resident*
- **Attributi non-resident**
 - esempio: l'attributo dati di un file
 - il loro contenuto viene memorizzato in una o più estensioni (*extent*) nel disco
 - il record del MFT contiene solo uno o più puntatori che indicano dove sono memorizzati i dati

NTFS: MFT

- **Esempi:**
 - se un file è particolarmente grande (o particolarmente frammentato, vedi lucidi seguenti):
 - il record MFT non riesce a contenere i puntatori ai blocchi dati
 - viene creato un “blocco indiretto” detto *base file record*
 - se un file è particolarmente piccolo:
 - i dati del file vengono inseriti direttamente nel record MFT
- **MFT è un file come tutti gli altri**
 - può essere collocato in qualunque parte del disco
 - può crescere a piacere, fino alla dimensione massima di 2^{48} record

NTFS: MFT

- **Gestione dei meta-dati**
 - i meta-dati di volume sono tutti contenuti in file
 - i primi 16 record del MFT sono riservati per questi file speciali
 - hanno nomi che iniziano con \$ per distinguerli dai file normali
- **\$Mft e \$MftMirr (posizione 0)**
 - contengono gli attributi principali del MFT, e in particolare i puntatori alla posizione del MFT
 - due copie per maggiore sicurezza
 - nota: è necessario un meccanismo di bootstrap per individuare il primo blocco del MFT; dal primo blocco è possibile trovare le informazioni necessarie per leggere il resto della MFT

NTFS: MFT

- **\$LogFile**
 - contiene la storia delle variazioni non ancora permanentemente registrate (consente undo e redo)
- **\$Volume**
 - contiene il nome del volume, la versione della formattazione e il bit di “spegnimento corretto”
- **\$AttrDef (Attribute definition table)**
 - quali tipi di attributi sono utilizzati nel volume e quali operazioni occorrono per gestirli
- **\$Root**
 - root directory

NTFS: MFT

- **\$Bitmap**
 - bitmap dei blocchi liberi o utilizzati
- **\$Boot**
 - file di bootstrap
- **\$BadClus**
 - lista dei cluster danneggiati
- **\$Secure**
 - informazioni di sicurezza
- **\$Upcase**
 - informazioni per la conversione upcase /lowcase
- **\$Extend**
 - varie estensioni, incluso il meccanismo di gestione delle quote
- **Più altri quattro record non utilizzati e riservati per uso futuro**

NTFS: File reference

- **Ogni file è individuato da un riferimento a 64 bit.**
 - i primi 48 bit (*file number*) rappresentano l'indice all'interno dell'MFT
 - gli ultimi 16 bit (*sequence number*) individuano quante volte l'elemento di MFT è stato riusato
- **Il sequence number**
 - è contenuto nello header di record
 - serve per non individuare erroneamente al posto di un file cancellato un file che ha riutilizzato quell'elemento di MFT

NTFS - Allocazione memoria secondaria

- **L'insieme di blocchi di un file**
 - viene suddiviso in uno o più sequenze di cluster "logicamente" contigui, eventualmente separate da "buchi"
 - in caso di buchi (sequenze di cluster con solo byte zero) si omette l'allocazione dei cluster contenuti nei buchi
- **Ogni sequenza:**
 - è composta da uno o più "run" di cluster fisicamente contigui
 - NTFS cerca di allocare i file nel modo più contiguo possibile

NTFS - Gestione delle directory

- **Per directory di piccole dimensioni**
 - la directory è composta da una sequenza lineare di elementi
- **Per directory di grandi dimensioni**
 - NTFS usa una struttura di B+tree come indice della directory stessa
 - tramite queste strutture dati è possibile gestire ricerche e inserimenti in modo efficiente
- **Note:**
 - per grandi gerarchie l'albero viene suddiviso (paginato) in più estensioni (caricabili singolarmente).
 - molte informazioni dell'MFT vengono duplicate per aumentare le performance.

NTFS: Compressione

- **NTFS ha anche funzionalità built-in per la compressione dei dati**
- **L'accesso ai file compressi è trasparente**
 - NTFS riconosce il file compresso e lo decomprime in lettura
 - effettua anche il prefetching se l'accesso è sequenziale
- **Meccanismo di compressione**
 - si suddivide il file in sezioni di 16 cluster
 - se la compressione di una sezione porta ad un guadagno in numero di cluster, la sezione viene memorizzata compressa
 - ogni sezione viene compressa in modo indipendente

NTFS: Fault-tolerance

- **NTFS gestisce RAID level 1 e RAID level 5**
- **Raid 1 (Disk mirroring):**
 - tutte le operazioni vengono fatte all'unisono su due volumi collocati in due dischi separati
- **Raid 5 (Disk striping)**
 - tollera il guasto di una singola unità su un numero arbitrario N. Viene perso 1/N della capacità totale per informazioni ridondanti

NTFS: Recovery da guasti

- **In NTFS**
 - le variazioni della MFT e degli altri file speciali avvengono all'interno di transazioni
 - le variazioni vengono registrate nell'area di log (che funziona come una coda circolare)
 - periodicamente (normalmente ogni 5 secondi) viene fatto un "checkpoint"
 - i log prima del checkpoint diventano inutili
- **In caso di guasto**
 - viene ripristinato il FS all'ultimo checkpoint valido
 - vengono eseguite nuovamente (redo) tutte le operazioni del log dopo il checkpoint

Windows 2000 - Networking

- **I protocolli sono caricati come driver**
 - in teoria potrebbero essere caricati dinamicamente
 - in pratica, è possibile che il sistema richieda di fare reboot in seguito ad una variazione
- **Protocolli utilizzati**
 - **Server Message Block (SMB)**
 - condivisione file e stampanti
 - **Network Basic I/O System (NetBIOS)**
 - protocollo standard per reti di PC negli anni 80.
 - **NetBios Estended User Interface (NetBEUI)**
 - protocollo standard di Win95 e WFW.

Windows 2000 - Networking

- **Protocolli utilizzati:**
 - **TCP/IP**
 - **Point-to-point tunneling protocol (PPTP)**
 - introdotto in NT 4.0, per interconnettere Windows attraverso IP e in generale Internet
 - **Novell IPX/SPX**
 - **Data-Link Control (DLC)**
 - per mainframe IBM e stampanti HP
 - **Appletalk**
 - condivisione file e stampanti del mondo Apple
 - richiede un apposito package.

Windows 2000 - Networking

• Meccanismi per IPC distribuito

- **Named Pipes**
 - meccanismo (connection oriented) per comunicare fra processi
 - vengono usati nomi secondo lo standard Uniform Naming Convention (UNC): \\server_name\share_name\x\y\z
- **Mailslot**
 - meccanismo connectionless
 - ammette broadcast per trovare componenti sulla rete
- **Windows Socket (WinSock)**
 - interfaccia compatibile con i socket UNIX
 - fornisce l'accesso a differenti protocolli che possono avere diversi schemi di indirizzamento

Windows 2000 - Networking

• Meccanismi per IPC distribuito

- **Remote Procedure Call (RPC)**
 - fornisce meccanismi di generazione stub/skeleton
 - gestisce marshalling/unmarshalling dei dati
 - può usare trasporto NetBIOS, WinSock su TCP-IP, named pipe o Lan Manager
- **Network Dynamic Data Exchange (DDE)**
 - è l'estensione in rete del meccanismo di IPC denominato DDE
 - la comunicazione ha luogo per mezzo di due pipe a senso unico

Windows 2000 - Networking

• Redirector e server

- in Windows 2000 è possibile accedere a file remoti come se fossero locali a patto che il sistema remoto abbia in funzione un **Server**
- un **Redirector** è l'oggetto (client-side) che redireziona la richiesta di I/O verso il **Server**

Windows 2000 - Networking

• Redirector e Server: interazione

- l'applicazione fa una richiesta di I/O specificando un UNC come nome file
- l'I/O manager riconosce che non è un file locale e chiama un oggetto che si chiama MUP (multiple universal-naming-convention provider)
- il MUP cerca (tramite broadcast) il redirector che può soddisfare la richiesta (una volta trovato fa cache della risposta)
- il redirector manda la richiesta al server
- il server riceve la richiesta e la elabora come se fosse locale
- il server invia il risultato al redirector remoto che infine consegna la risposta al processo.

Windows 2000 - Networking

- **Dominio - Descrizione**
 - un Dominio NT è un insieme di workstation che condividono politica di sicurezza e data base degli utenti
- **Architettura**
 - un server funziona da *primary domain controller*
 - ci possono essere altri server che funzionano da backup del server principale
 - consentono migliori performance perché le richieste sono ripartite fra tutti i server (primary e non)

Windows 2000 - Networking

- **Quattro diversi modelli:**
 - *single-domain model*
 - *master-domain model*
 - il master domain è trusted da tutti i sottodomini
 - *multiple-master domain model*
 - come il precedente ma ci sono molteplici master-domain, tutti trusted fra loro
 - *multiple-trust model*
 - non ci sono "master" tutti i domini sono trusted fra loro

Windows 2000 - Networking

- **Name Resolution in NT**
 - NT usa molteplici tecniche per la Name Resolution.
- **Windows Internet Name Service (WINS)**
- **Broadcast name resolution**
- **DNS (quello di Internet)**
- **"Hosts" file**

Windows 2000 - Networking

- **WINS**
 - può gestire una allocazione dinamica degli indirizzi IP.
 - il protocollo Dynamic Host Configuration Protocol (DHCP) aggiorna il database di WINS ad ogni connessione di un client
 - si dice che l'indirizzo Ip viene "noleggiato" (leased) alla stazione
 - se il noleggio non viene rinnovato l'indirizzo viene riutilizzato
- **DHCP**
 - Cliente: discover message (in Broadcast)
 - Server: offer message contenente la configurazione (e l'indirizzo IP)
 - Cliente: request message contenente la richiesta della configurazione scelta