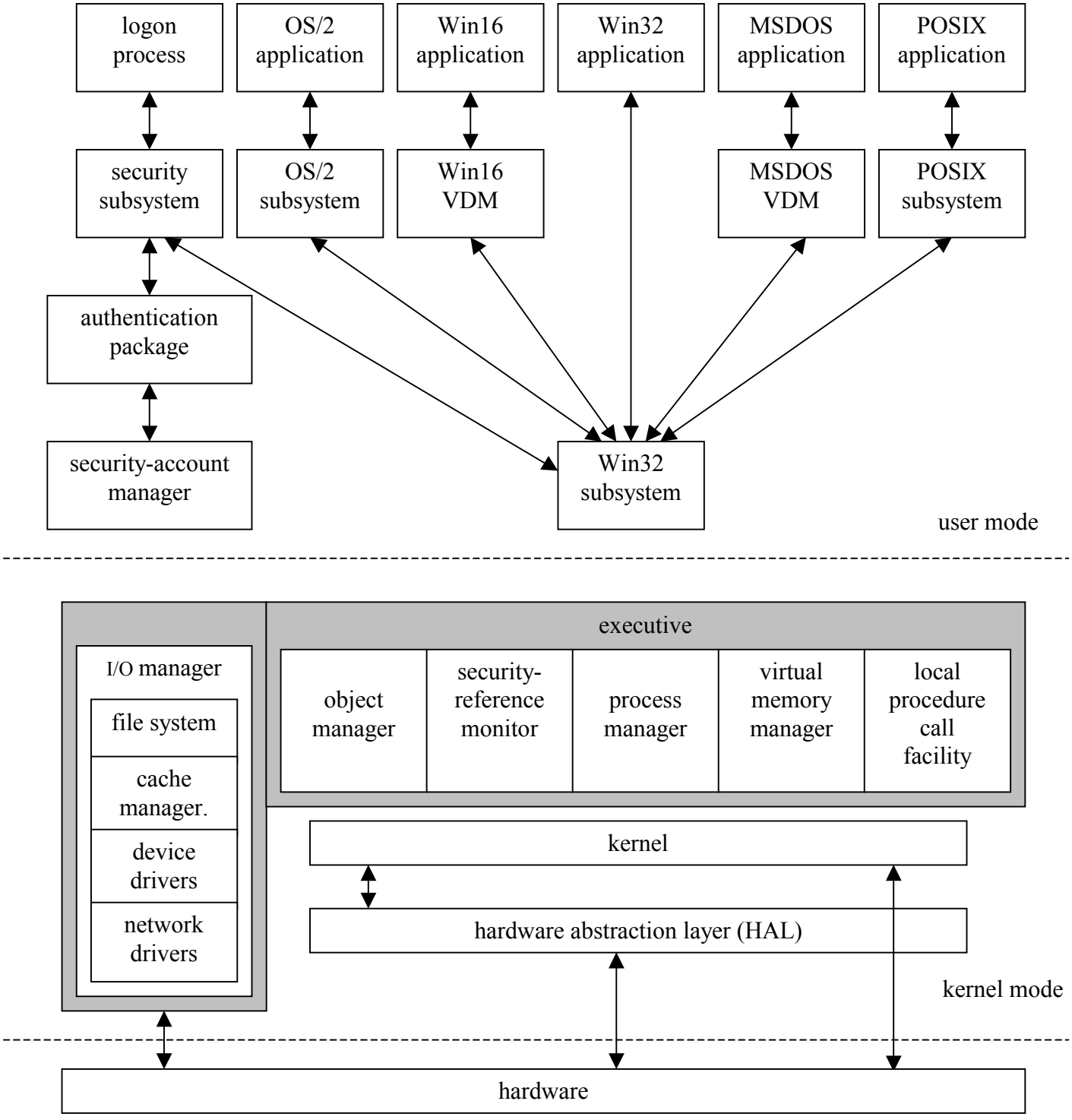
WINDOWS NT

Diagram warstw systemu Windows NT



Hardware abstraction layer (HAL)

- warstwa oprogramowania ukrywająca różnice sprzętowe przed wyższymi warstwami systemu operacyjnego — dostarcza interfejsu używanego przez jądro, zarządcę (ang. *executive)* i sterowniki urządzeń

Jądro

- używa dwóch rodzajów obiektów — *dispatcher* (np. zdarzenia, mutex-y, semafory, wątki czy zegary) i *control* (np. asynchroniczne wywołania procedur, przerwania, procesy czy profile)
- procesy i wątki — każdy proces zawiera jeden lub więcej wątków
- stany wątków → gotowy, bezczynny, działający, oczekujący, nowy i zakończony
- strategia karuzelowa, kolejka wielopoziomowa ze sprzężeniem zwrotnym
- planista używa 32 poziomów priorytetów do ustalenia porządku wątków — z każdym poziomem związana jest jedna kolejka — wybierany jest wątek o najwyższym priorytecie, gotowy do wykonania
- priorytet wątku jest zmniejszany zawsze po wykorzystaniu kwantu czasu procesora a zwiększany gdy proces wychodzi ze stanu oczekiwania
- procesy z klasy *real-time* mają większe priorytety i prawo wywłaszczania procesów klasy *variable*
- procesy działają w trybie użytkownika lub w trybie jądra

- wyjątki i przerwania sprzętowe i programowe — „błękitny ekran śmierci” oznacza wyjątek w trybie jądra dla którego nie ma funkcji obsługi
- 32 poziomy przerwań — 8 używanych przez jądro i 24 reprezentujące przerwania sprzętowe przez HAL

Zarządca

- dostarcza zbiór usług, których mogą używać podsystemy odpowiedzialne za poszczególne środowiska — zarządzanie obiektami, zarządzanie pamięcią wirtualną, zarządzanie procesami, umożliwienie lokalnego wywoływania procedur, zarządzanie I/O i monitorowanie bezpieczeństwa

Windows NT to system obiektowy — używa obiektów dla wszystkich swoich encji i serwisów (np. obiekty katalogów, linków symbolicznych, zdarzeń, procesów i wątków)

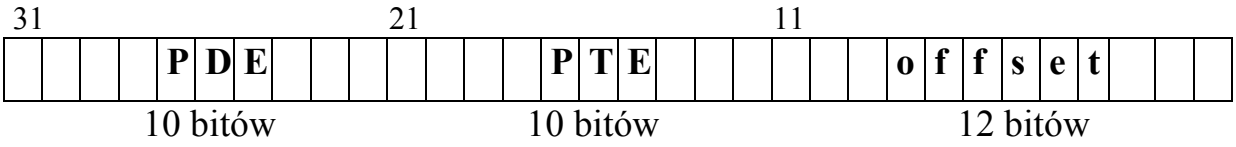
Zarządca obiektów

- obiekt może być używany tylko po otrzymaniu uchwytu — interfejsu do obiektu
- idealne miejsce do sprawdzania uprawnień i bezpieczeństwa — każdy użytkownik w momencie zalogowania się do systemu otrzymuje *żeton dostępu* (zawierający dane identyfikujące) umożliwiający korzystanie z pewnego zbioru obiektów
- utrzymuje informacje o tym jakie obiekty są używane przez jakie procesy — każdy obiekt zawiera licznik odwołań (ilość uchwytów) i licznik

referencji (system często używa wskaźników zamiast uchwytów); gdy któryś z liczników zmaleje do zera obiekty (tymczasowe) są usuwane z pamięci
- obiekty stałe reprezentują encje fizyczne, np. napędy dyskowe itp.
- do manipulacji obiektami służy standardowy zbiór metod np. **create, open, close, delete** itp.
- obiekty mogą posiadać nazwy — przestrzeń nazw obiektów jest globalna i umożliwia ich dzielenie
- każdy proces utrzymuje *tablicę obiektów* zawierającą m.in. prawa dostępu do obiektu

Zarządca pamięci wirtualnej
- wymagamy od sprzętu umiejętności odwzorowywania pamięci wirtualnej do fizycznej, mechanizmu stronicowania i możliwości dzielenia ramek pamięci fizycznej
- stronicowanie na żądanie, strona wielkości 4KB, strony przydzielone procesom, które nie są w pamięci fizycznej przechowywane są w *pliku wymiany*
- adresy 32-bitowe, zatem do 4GB pamięci wirtualnej dla każdego procesu — górne 2GB, identyczne w każdym procesie są używane tylko w trybie jądra
- dwustopniowa alokacja pamięci — rezerwacja przestrzeni adresowej procesu + potwierdzenie alokacji przez przydział przestrzeni w pliku wymiany
- przestrzeń wirtualna nowo tworzonego procesu jest dostępna dla procesu rodzica — pamięć

reprezentowana przez obiekty, które można dzielić (za pomocą uchwytów)
- segmenty pamięci mogą być dzielone przez wiele procesów
- każdy proces ma *katalog stron* (ang. *page directory)*, który zawiera 1024 pozycje (PDE) rozmiaru 4B każda; każda taka pozycja wskazuje na *tablicę stron*, która zawiera 1024 pozycje tablicy stron (PTE) rozmiaru 4B każda; każda PTE wskazuje na 4KB ramkę pamięci fizycznej (całkowity rozmiar wszystkich tablic stron procesu to 4MB)
- 32-bitowy adres wirtualny jest dzielony na trzy części:



- wskaźnik do pojedynczego bajta w pamięci fizycznej składa się z 20 bitów z PTE (która ma 4B) i 12 bitów przesunięcia (z adresu wirtualnego) — pozostałe 12 bitów w PTE służy do opisania atrybutów (RO, RW) i stanu strony (poprawna, wyzerowana, wolna, oczekująca, zmodyfikowana lub zła)
- dla stron dzielonych PTE w każdym z procesów wskazuje na pozycję prototypową zamiast bezpośrednio na ramkę — dzięki temu każda zmiana na stronie wymaga poprawienia zapisu w jednym miejscu

- zarządca pamięci wirtualnej utrzymuje bazę danych odwzorowań **strona ↔ ramka;** dla każdej ramki pamięci fizycznej ma wskaźnik do PTE zawierającej wskaźnik do danej ramki
- błąd braku strony powoduje sprowadzenie strony do pierwszej wolnej ramki (wziętej z listy) oraz sprowadzenie kilku stron sąsiednich (przy założeniu lokalności odwołań pozwala to ograniczyć liczbę braków stron)
- gdy nie ma wolnych ramek używany jest algorytm FIFO do zabrania ramek procesom, które używają więcej stron niż wynosi minimalny rozmiar ich zbioru roboczego
- domyślny rozmiar zbioru roboczego to 30 ramek, ale system monitoruje błędy braku strony procesów, żeby określić jego prawidłowy rozmiar

Zarządca procesów

- dostarcza usług tworzenia, usuwania i używania wątków oraz procesów
- nie posiada informacji o zależnościach rodzic-dziecko ani o hierarchii procesów — te dane są przeniesione do systemów obsługujących poszczególne podsystemy (np. Win32)

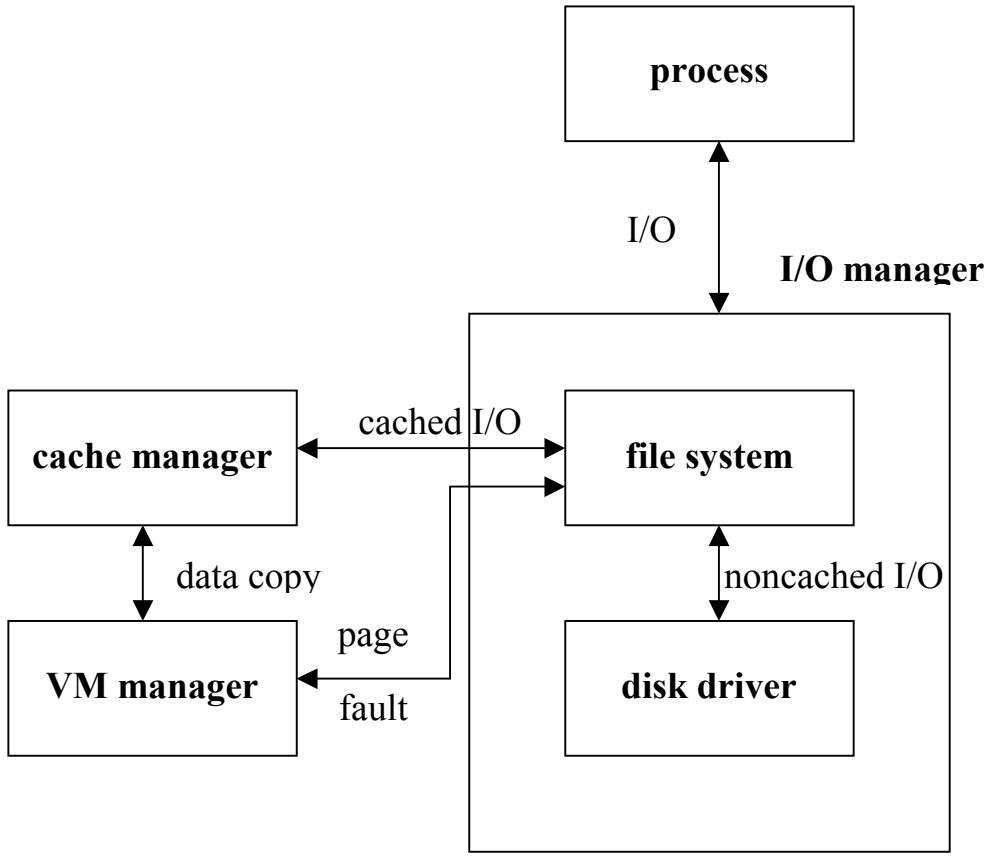
Lokalne wywoływanie procedur (LPC)

- mechanizm przekazywania wiadomości używany do odwoływania się do usług poszczególnych podsystemów (architektura klient-serwer)

Zarządca wejścia-wyjścia

- utrzymuje informację o dołączonych systemach plików i zarządza buforami podręcznymi (na żądania operacji I/O)
- współpracuje z zarządcą pamięci wirtualnej (plik wymiany) i kontroluje nadzorcę pamięci podręcznej CM (obsługującego **cały** system I/O)
- umożliwia zarówno synchroniczne jak i asynchroniczne przesyłanie danych
- tłumaczy otrzymane żądania do standardowej postaci (zwanej *I/O request packet* IRP), a następnie przekazuje je do sterowników odpowiednich urządzeń w celu zrealizowania — po zakończeniu operacji dostaje IRP od odpowiedniego sterownika
- rozmiar pamięci podręcznej zmienia się dynamicznie (w zależności od ilości wolnej pamięci)
- zarządca pamięci wirtualnej rezerwuje do połowy górnej pamięci wirtualnej na pamięć podręczną systemu
- nadzorca pamięci podręcznej mapuje pliki na tę przestrzeń adresową i wykorzystuje działanie zarządcy pamięci wirtualnej do obsługi operacji wejścia-wyjścia na plikach
- pamięć podręczna jest podzielona na bloki 256KB — każdy blok zawiera zmapowany w pamięci fragment pliku
- każdy blok pamięci podręcznej jest opisany przez blok kontrolny adresu wirtualnego (VACB)

- dla każdego otwartego pliku nadzorca pamięci podręcznej utrzymuje tablicę indeksów VACB
- każde żądanie otrzymane przez zarządcę I/O jest kierowane do nadzorcy pamięci podręcznej, chyba że jest to specyficzne żądanie **„bez pamięci podręcznej”** skierowane przez zarządcę pamięci wirtualnej na skutek błędu braku strony



- w celu poprawy wydajności nadzorca pamięci podręcznej utrzymuje krótką historię przeprowadzonych operacji i stara się przewidzieć przyszłe
- typowe operacje I/O przeprowadzane są blokami wielkości 64KB (16 stron)

- domyślnym zachowaniem nadzorcy pamięci podręcznej jest dokonywanie zapisu pamięci na dysk co 4-5 sekund (chyba, że proces zadecyduje inaczej)

Podsystemy poszczególnych środowisk

- podsystemy środowiskowe są procesami poziomu użytkownika korzystającymi z usług dostarczanych przez główny system Win32 w celu uruchamiania programów przeznaczonych dla innych systemów operacyjnych np. 16-bitowych Windows, MS-DOS, POSIX lub pewnych aplikacji 16-bitowego OS/2
- używają mechanizmu LPC do komunikacji, system dba o to aby odpowiednie wywołania trafiały do odpowiednich podsystemów

Win32

- główne środowisko pracy — uruchamia wszystkie procesy, decyduje czy aplikacja powinna zostać wykonana w innym środowisku i inicjuje takie wykonanie
- zarządza klawiaturą, myszką i środowiskiem graficznym
- każdy proces ma własną kolejkę wejściową — źle działające procesy nie mogą blokować systemu
- wszystkie obiekty są sprawdzane przed użyciem

MS-DOS

- *virtual dos machine (VDM)* — aplikacja Win32 emulująca instrukcje procesora Intel 486, MS-DOS ROM BIOS, programowe przerwanie 21; bazuje na MS-DOS 5.0, dostarcza programom co najmniej 620KB pamięci
- aplikacje dos-owe, które potrzebują bezpośredniego dostępu do urządzeń (fizycznych) nie będą działać

Win16

- środowisko dostarczane również przez VDM + dodatkowe oprogramowanie *Windows on Windows*
- aplikacje Win16 polegające na wewnętrznej strukturze 16-bitowego systemu mogą nie działać prawidłowo
- tylko jedna aplikacja może działać w jednym czasie, aplikacje są jednowątkowe i zajmują ten sam obszar przestrzeni adresowej oraz dzielą kolejkę wejściową

POSIX

- środowisko dla aplikacji przestrzegających standardu POSIX.1
- aplikacje mogą być uruchamiane przez Win32 lub przez inne aplikacje POSIX
- aplikacje mają dostęp do dowolnego systemu plików; na plikach obowiązują prawa dostępu i struktura katalogów takie jak w Unixie

OS/2

- pierwotnie miało to być domyślne środowisko dla WindowsNT
- dostarcza tylko ograniczonej funkcjonalności OS/2 — tylko „znakowe” aplikacje OS/2 1.x mogą działać na NT i tylko na architekturze Intel
- prawdziwe aplikacje OS/2 mogą działać przy użyciu środowiska MS-DOS

Podsystemy logowania i bezpieczeństwa

- umożliwia identyfikację użytkowników na podstawie hasła
- generuje żetony dostępu do zasobów reprezentujące użytkowników systemu

System plików

- ewolucja: FAT16 (wewnętrzna fragmentacja, max. 2GB, brak ochrony) → FAT32 (rozwiązane problemy z rozmiarem i fragmentacją, ale kłopoty z wydajnością) → NTFS (zawiera rozwiązania wspomagające odzyskiwanie danych, zapewnienie bezpieczeństwa danych, odporność na błędy, duże pliki i systemy plików, zwielokrotnione strumienie danych, wsparcie dla UNICODE, kompresja)

NTFS

- wolumen jako podstawowa encja NTFS — może zajmować część dysku (logiczną partycję), cały dysk lub nawet rozciągać się na kilka dysków

- nie działa na pojedynczych sektorach dyskowych, ale na klastrach (ang. *cluster*), które składają się z kilku sektorów (potęga 2) — rozmiar klastra jest konfigurowany w momencie tworzenia systemu plików NTFS (typowe rozmiary: 1 sektor dla wolumenów do 512MB, 1KB dla wolumenów do 1GB, 2KB dla wolumenów do 2GB, 4KB dla wolumenów większych)
- logiczne numery klastrów (LCNs) jako adresy dyskowe

Pliki

- plik nie jest prostym strumieniem bajtów (jak w Unixie), ale obiektem z wewnętrzną strukturą składającym się z atrybutów — każdy atrybut pliku jest niezależnym strumieniem bajtów, który może być tworzony, usuwany, odczytywany i zapisywany
- plik zawiera atrybuty standardowe (np. nazwa(-y), czas utworzenia, deskryptor kontroli dostępu) + atrybuty charakterystyczne dla poszczególnych rodzajów plików (np. katalogi zawierają atrybuty implementujące indeks nazw plików w katalogu)
- najbardziej powszechne (zwykłe) pliki zawierają atrybut „nienazwany” zawierający dane pliku
- atrybuty mogą różnić się rozmiarem
- każdy plik jest opisany przez jeden lub więcej rekordów w tablicy przechowywanej w specjalnym pliku zwanym *master file table (MFT)* — plik ten ma kopię (MFT2) umożliwiający odzyskanie danych utraconych przez uszkodzenie MFT

- małe atrybuty, tzw. rezydentne są przechowywane w MFT
- duże atrybuty (nierezydentne), takie jak „nienazwane" atrybuty z danymi są przechowywane w jednym lub kilku ciągłych obszarach na dysku, a wskaźniki to każdego z tych obszarów są przechowywane w rekordach MFT
- dla pliku z dużą ilością atrybutów (np. bardzo fragmentowanego) rozmiar MFT może być za mały (nie zmieszczą się wszystkie wskaźniki) — taki plik jest opisany przez rekord zwany *bazowym rekordem pliku*, który zawiera wskaźniki do rekordów, które zawierają dodatkowe wskaźniki i atrybuty
- każdy plik na wolumenie NTFS ma unikalny identyfikator — 64-bitowy numer referencyjny (48 bitów to numer rekordu w tablicy MFT + 12 bitów to numer sekwencyjny zwiększany podczas każdego użycia rekordu tablicy MFT)

Katalogi

- hierarchiczna, drzewiasta struktura katalogów
- każdy katalog używa struktury danych zwanej *B+ drzewem* do przechowywania indeksu plików w katalogu
- każda pozycja w katalogu zawiera nazwę pliku i jego numer referencyjny oraz znacznik czasowy ostatniej zmiany i rozmiar pliku (wzięte z rekordu MFT)
- wszystkie metadane o wolumenie NTFS są przechowywane w specjalnych plikach:

- ✓ $mft — MFT; zawiera też informacje o sobie
- ✓ $mftmirr — MFT2; plik zawierający kopię pierwszych 16 pozycji z MFT
- ✓ $logfile — zapis transakcji w systemie plików; używany do reperacji systemu plików
- ✓ $volume — zawiera nazwę wolumenu, wersję NTFS, która posłużyła do utworzenia wolumenu i bit poprawności
- ✓ $attrdef — jakiego typu atrybuty są używane na wolumenie i jakie operacje mogą zostać przeprowadzone na każdym z nich
- ✓ $bitmap — które klastry na wolumenie są przypisane do plików, a które są wolne
- ✓ $boot — zawiera kod startowy dla NT (jeżeli wolumen jest startowy); musi być umieszczony w szczególnym miejscu na dysku, aby mógł zostać odnaleziony przez program ładujący
- ✓ $badclus — zawiera informację o wszystkich złych klastrach na wolumenie; używany do reperacji systemu plików
- ✓ $upcase — zawiera informację o konwersji nazw plików do znaków UNICODE-u