NTFS – omówienie systemu plików

autor: Julian Krzemiński
email: J.Krzeminski@students.mimuw.edu.pl

Źródła:

Microsoft Windows Internals - Mark E. Russinovich, A. Solomon

NTFS - system plików, którego celem było sprostanie nowym wymaganiom

Przed systemem plików zaczęto stawiać nowe wymagania, którym poprzedni system plików FAT nie był w stanie sprostać.

Wymagania te były na tyle skomplikowane, że nie zdecydowano się na rozszerzenie systemu FAT tylko napisanie zupełnie nowego systemu plików. System ten nazwano NTFS (New Technology File System).

Nowe cechy miały nie tylko zaspokoić oczekiwania biznesowe, ale i wspomóc zwykłych użytkowników w ich codziennej pracy.

Główne cele, które postawiono przed nowym systemem plików to:

- *zminimalizować utratę i rozspójnienie danych w przypadku awarii sprzętu, braku zasilania, awarii systemowych - NTFS zapewnia integralność metadanych (czyli danych opisujących plik) w przypadku awarii*
- *ochrona danych przed nieautoryzowanym dostępem - NTFS ma zintegrowany model bezpieczeństwa dzięki któremu użytkownik może chronić swoje dane*

NTFS – cechy ogólne

- **obiektowy systemu plików** – otwarty plik to obiekt o pewnych atrybutach, atrybutem pliku oprócz np. nazwy, uprawnień są dane pliku. NTFS pozwala na definiowanie kolejnych atrybutów – dzięki czemu np. możemy utworzyć plik o wielu strumieniach danych

  (multiple data streams). Z każdym obiektem system wiąże deskryptor uprawnień – pozwala na szczegółowe zdefiniowanie uprawnień użytkowników względem akcji na obiekcie.

- **Bezpieczeństwo danych (Security)**

  W NTFS wprowadzono obiektowy model danych. Otwarty plik zaimplementowany jest jako obiekt z którym związany jest desktyptor bezpieczeństwa (security descriptor). Deskryptor ten jest przechowywany na dysku jako część pliku. Zanim dowolny proces otrzyma dowiązanie do dowolnego obiektu (np. obiektu pliku), weryfikowany są uprawnienia procesu do danego obiektu na podstawie deskryptora bezpieczeństwa.

  Proces nie mający odpowiednich uprawnień nie uzyska dostępu do obiektu.

- **nazwy** plików, folderów, wolumenów w Unicode – pozwala na znaki diaktryczne, nazwa może zawierać maksymalnie 255 znaków
- **indeksowanie** plików i katalogów po nazwie oraz dodatkowo NTFS indeksuje pliki po atrybutach stąd szybko można wyszukać pliki spełniające pewne określone kryteria np. wszystkie pliki o rozmiarze pomiędzy 1MB a 12MB.
- **Quota** – administrator ma możliwość śledzenia zużycia powierzchni dyskowej oraz wprowadzania ograniczeń dyskowych dla użytkowników – NTFS zlicza logiczny rozmiar plików tzn. bez względu na to czy kompresuje dane czy nie do sumy rozmiarów liczy się rozmiar nieskompresowanych danych
- **Szyfrowanie** danych – przezroczyste dla aplikacji – użytkownik może zażądać szyfrowania danych – wtedy dla aplikacje działających z odpowiednimi uprawnieniami NTFS sam odszyfruje dane. Plik zostanie zaszyfrowany gdy ustawimy odpowiedni atrybut pliku. Szyfrowanie odbywa się poprzez klucze prywatne i publiczne. To na co należy uważać to na kopie plików tworzone przez programy (np. Editplus, Word) – kopie te oczywiście nie są szyfrowane ale tworzone są w tym samym folderze co plik źródłowy – dlatego dobrze jest szyfrować cały katalog nawet gdyby miało to oznaczać katalog z jednym plikiem.
- **Kompresja** danych – przezroczysta dla użytkownika lub aplikacji – NTFS sam dba o kompresje i dekompresje danych – katalogi kompresowane są rekurencyjnie.
- **Pliki rzadkie** (sparse file) – plik, często bardzo duży, zawierający niewielką liczbę danych niezerowych. Jeśli plik oznaczony jest jako „sparse file” system alokuje dla niego miejsca na dysku tylko na dane niezerowe. Kiedy aplikacja czyta niezaalokowany obszar pliku, NTFS generuje bufor wypełniony zerami o odpowiedniej długości i przekazuje go do aplikacji.
- **Transakcyjne** wykonywanie operacji We / Wy z pełnym prowadzeniem dziennika operacji na metadanych (strukturze NTFS) – pozwala na odtworzenie spójnej struktury systemu plików
- Przezroczysta dla użytkownika podmiana uszkodzonych sektorów
- **Twarde dowiązania** – dodanie nowych atrybutów filename wskazujących na nowe dowiązania

NTFS - Struktura systemu

- Najmniejszą logiczną jednostką na dysku jest klaster – najczęściej jest to ośmiokrotność wielkości sektora dyskowego czyli 4KB.

NTFS odwołuje się do klastrów pliku poprzez VCN (Virtual Cluster Numbers) – numerowane od 0 do m (nie oznacza to że obszar na dysku jest ciągły) – te numery z kolei są mapowane na LCN (logical cluster numbers) – które można już zmapować na adres fizyczny.

- Wszelkie dane składowane są na dysku w postaci plików wliczając w to indeksy, bitmapy, metadane i dane.

  Pozwala to na szybkie zlokalizowanie i łatwe utrzymywanie danych, każdy z plików chroniony jest własnym deskryptorem uprawnień.

  Trzonem systemu NTFS jest **MFT** – jest to tablica jednokilobajtowych rekordów. Logicznie każdemu plikowi odpowiada jeden rekord wliczając w to plik z MFT.

Struktura wolumenu:

BOOT SECTOR	MFT	SYSTEM FILES	DATA

Boot sector zawiera kod potrzebny do wystartowania systemu.

MFT – tablica rekordów

Pierwsze 16 wpisów w MFT jest zarezerwowane na metadane opisujące wolumen. Nazwy tych plików zaczynają się od znaku $.

Pliki te to właśnie pliki systemowe (SYSTEM FILES)

Figure 12-22. NTFS data structures

[View full size image]

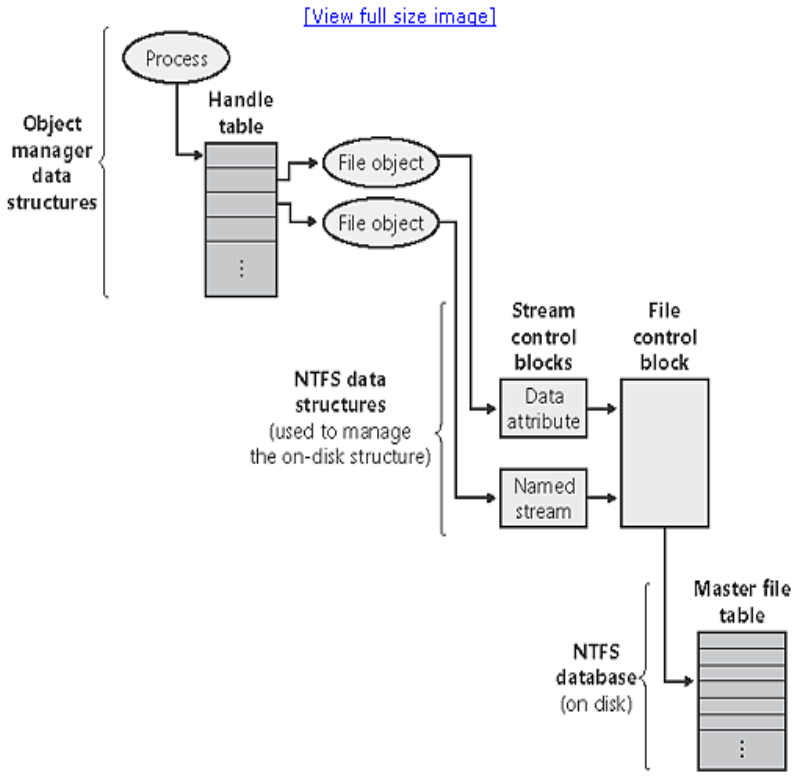
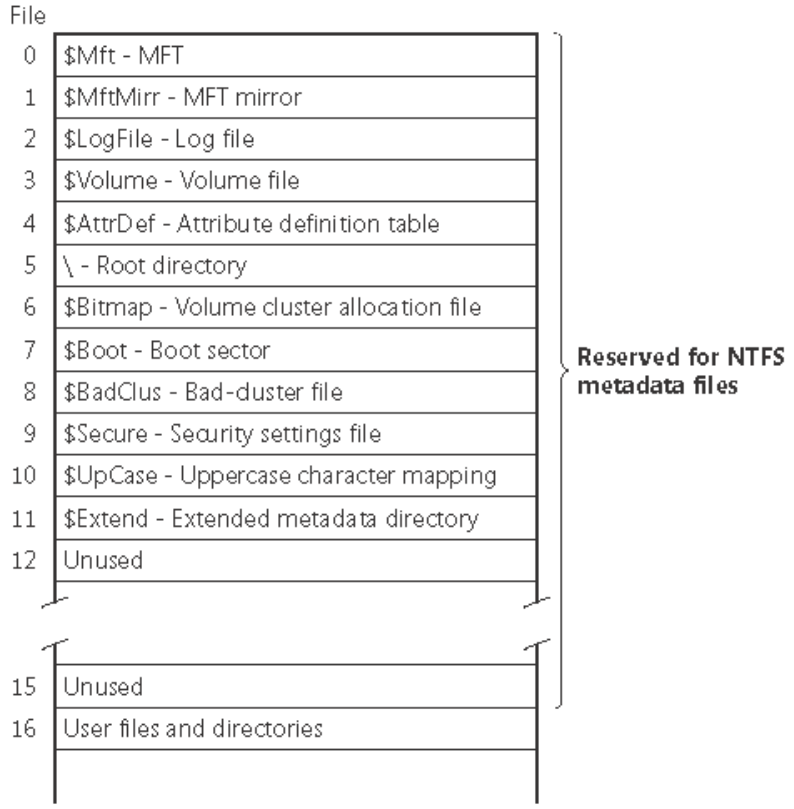
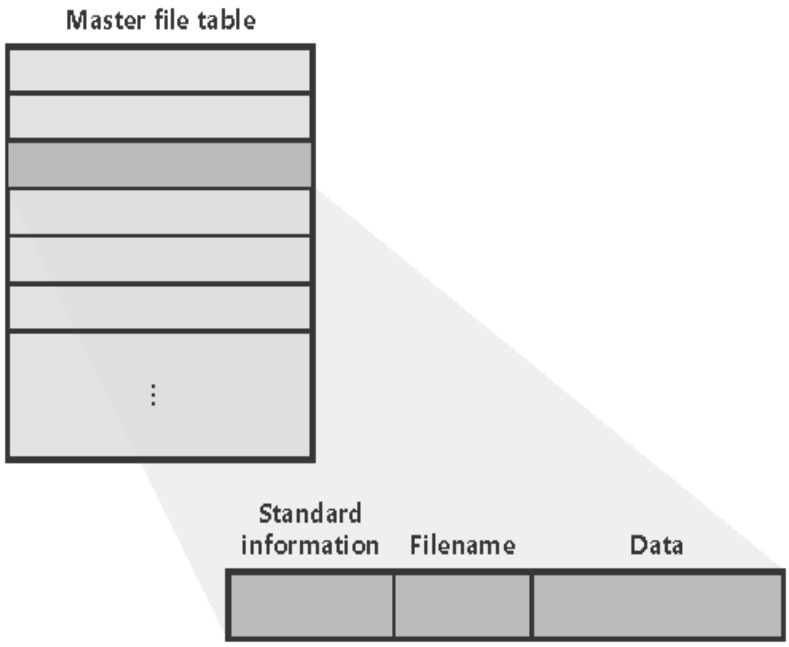


Figure 12-24. File records for NTFS metadata files in the MFT



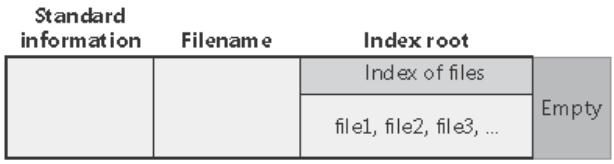
Dla małych plików – ich cała zawartość mieści się w rekordzie w tablicy MFT:

Figure 12-26. MFT record for a small file



Analogicznie dla niewielkiego katalogu:

Figure 12-30. MFT file record for a small directory



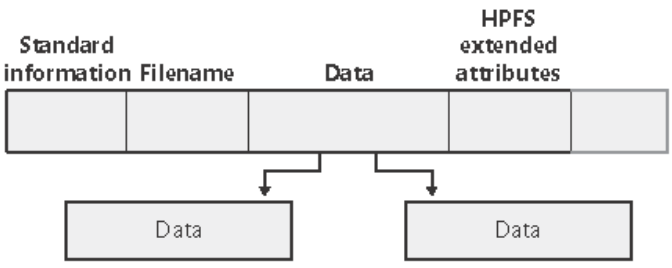
Czym są dane pliku?

NTFS definiuje pary: atrybut / wartość, tak więc zawartość pliku to po prostu jeden z atrybutów pliku o nazwie Data. Pozostałe atrybuty to :

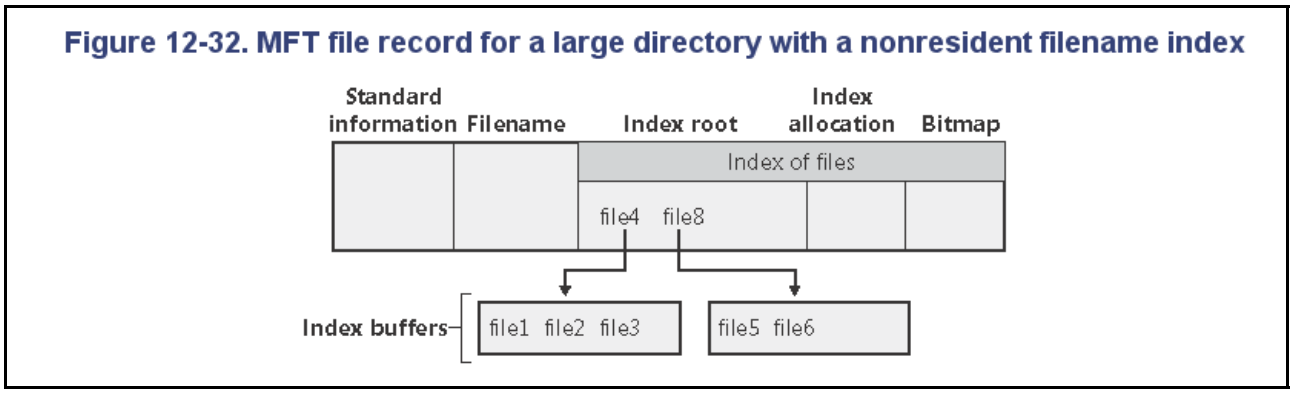
- nazwa pliku
- deskryptor uprawnień
- indeksy (tylko dla katalogów)
- id obiektu
- EFS – na potrzeby szyfrowania, wersja klucza od odszyfrowania danych plus lista użytkowników mających dostęp do danych
- informacje na temat wolumenu
- standardowe informacje (tylko-do-odczytu, archiwum, stemple czasowe(stempel utworzenia , stempel ostatniej modyfikacji), licznik twardych odnośników)

Jednakże częsta jest sytuacja, gdy plik nie mieści się w rekordzie w MFT – wtedy w rekordzie w MFT jest pula wskaźników do danych, jeżeli plik jest na tyle duży że w rekordzie nie ma wystarczająco miejsca na pule wskaźników, to w rekordzie jest wskaźnik do obszaru zawierającego pulę wskaźników.

Figure 12-31. MFT file record for a large file with two data runs



Analogicznie katalog może zawierać na tyle wiele plików, że niemożliwe będzie zbudowanie indeksu (B+ drzewa) w rekordzie.



B+ drzewo to drzewo mające dowiązania do danych jedynie w liściach, dodatkowo na liściach jest fastryga co pozwala na niezwykle szybkie wyszukiwanie zakresowe.

NTFS - Rozwiązania dla działania w warunkach awaryjnych, odporność systemu (Recoverability)

System odtwarzania bazuje w dużym stopniu na systemach odtwarzania spotykanych w systemach bazodanowych. W NTFS wprowadzono koncepcje transakcyjności operacji. Wszelkie operacje na danych wykonywane są w transakcjach. Podstawową cechą transakcji jest atomowość jej wykonania tzn. albo wykonane zostaną wszystkie operacje składające się na pojedynczą transakcję lub nie zostanie wykonana żadna z operacji (może to wymagać wycofania (rollback) pewnej liczby operacji w przypadku przerwania transakcji).

Transakcyjność operacji gwarantuje poprawność i integralność metadanych czyli struktury systemu plików dzięki czemu system zawsze będzie w stanie spójnym.

NTFS traktuje każdą operację We / Wy , która zmienia strukturę systemu plików, zmienia strukturę katalogu, modyfikuje plik, alokuje miejsce dla nowego pliku jako atomową transakcję. System ten gwarantuje, że transakcja będzie ukończona lub w przypadku awarii wycofana.

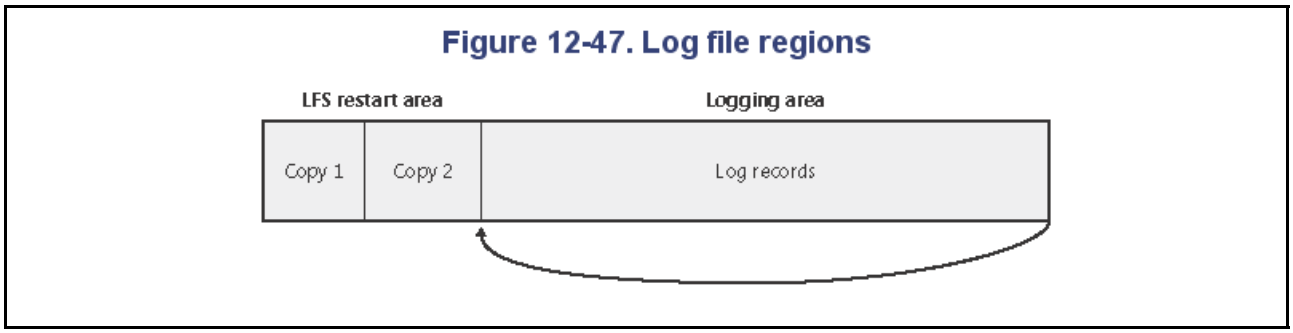
Zasadniczą techniką wspierającą odporność na awarie jest prowadzenie dziennika wszystkich operacji wykonanych na danych. Istotną rolę pełnią tu punkty kontrolne (checkpoints) - dzięki którym unikamy sytuacji analizy dziennika z odległej przeszłości.

Ogólnie dzienniki dzielimy na trzy kategorie:

- undo (unieważnienie)
- redo (powtarzanie)
- undo / redo

Dziennik zastosowany w NTFS jest typu undo / redo (unieważnienie / powtarzanie). Oznacza to większą złożoność przy odtwarzaniu danych po awarii lecz znacznie większą elastyczność podczas działania w trybie normalnym.

Dziennik to plik:



Ogólnie stosując tylko dzienniki undo musimy zapisywać dane na dysk zaraz po zakończeniu transakcji – nie możemy zapisać do dziennika rekordu zakończenia transakcji jeśli dane nie są fizycznie na dysku. Odtwarzanie przy dziennikach undo polega wycofaniu wszystkich nie zakończonych transakcji zapisanych do dziennika.

Natomiast stosując jedynie dzienniki redo zmodyfikowane bloki musimy trzymać w buforach aż do zatwierdzenia transakcji – nie możemy zapisać ich na dysk jeżeli nie wiemy czy transakcji zakończy się pomyślnie. Odtwarzanie w dziennikach redo polega na powtórzeniu wszystkich zakończonych transakcji.

Rozsądnym rozwiązaniem tzn. rozwiązaniem, które jest znacznie bardziej elastyczne w zakresie kolejności wykonywania czynności jest stosowanie dzienników undo / redo.

Jedyna reguła w dziennikach undo / redo to:

- Zanim na dysku zapiszemy nową wartość elementu X, spowodowanej działaniem pewnej transakcji T, najpierw należy na dysk wprowadzić zapis <T, X, v, w> gdzie v – stara wartość, w – nowa wartość

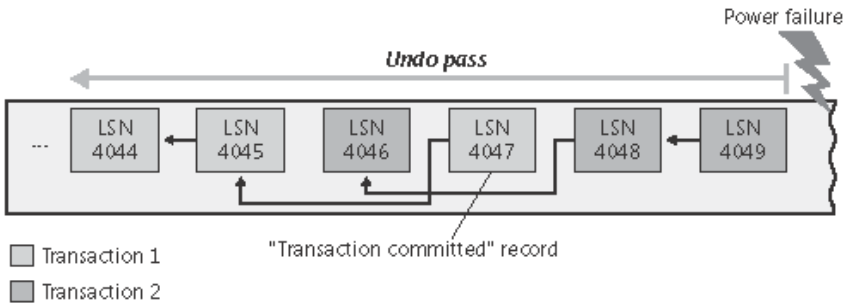
Nie ma narzutu na operację commit – możemy ją wprowadzić do dziennika zarówno po zapisaniu danych na dysk lub przed.

Odtwarzanie polega na :

- powtórzeniu wszystkich transakcji zatwierdzonych, zaczynając od najwcześniejszej
- unieważnieniu wszystkich transakcji niezakończonych, zaczynając od najpóźniejszej

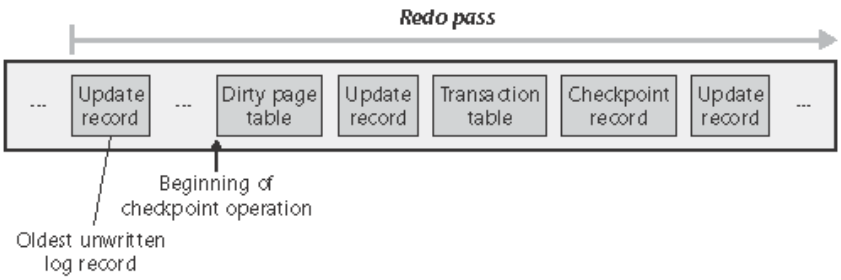
Odtwarzanie undo:

Figure 12-52. Undo pass



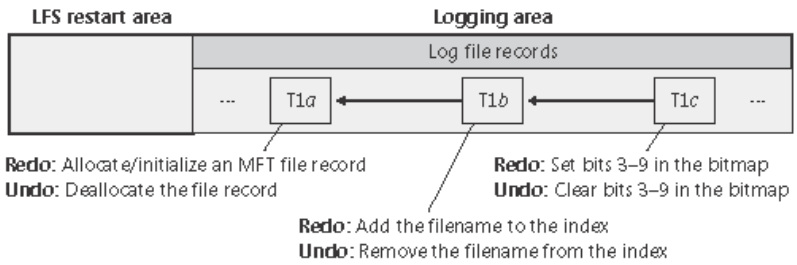
Odtwarzanie redo:

Figure 12-51. Redo pass



Przebieg odtwarzania :

Figure 12-48. Update records in the log file



Punkty kontrolne (Checkpoints)

Punkt kontrolny w dzienniku jest zapisem, dzięki któremu nie musimy przy odtwarzaniu czytać dziennika z odległej przeszłości.

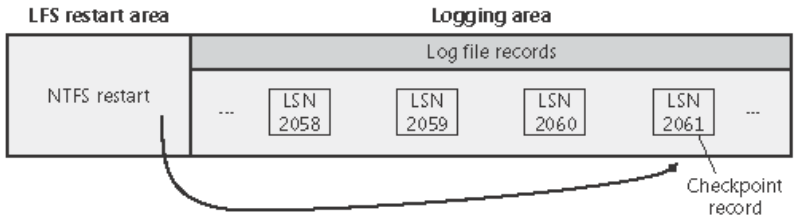
Punkty kontrolne różnią się w zależności od dziennika zastosowanego ale w NTFS gdzie zastosowano dziennik undo / redo znaczenie punktów kontrolnych jest następujące:

I. Do dziennika zapisujemy <START CKTP l > gdzie l to lista aktywnych transakcji w danym momencie.

II. Zapisujemy na dysk wszystkie brudne bufory (te z zatwierdzonych ale i z niezatwierdzonych transakcji)

III. Zapisujemy do dziennika <END CKTP>.

Figure 12-49. Checkpoint record in the log file Checkpoint record



System NTFS wysyła do dziennika zapis o punkcie kontrolnym co 5 sekund.

Najpierw system wykonuje odtwarzanie z powtarzaniem, następnie z unieważnieniem – zauważmy, że nie ma znaczenia, które odtwarzanie wykonamy najpierw bo gdy transakcje są izolowane to żadne dwie nie będą zmieniać tego samego elementu jednocześnie.

Przy odtwarzaniu **z powtarzaniem** system wykonuje:

- jeżeli czytając dziennik od końca napotka na wpis <End CKTP> to powtarza jedynie zatwierdzone transakcje rozpoczęte po tym dokonaniu wpisu <Start CKTP> (odpowiadającemu dokładnie temu <End CKTP>)

oraz te transakcje będące na liście aktywnych transakcji.

- Jeżeli czytając dziennik od końca napotka na wpis <Start CKTP> to jest on ignorowany tzn. szukamy pierwszego rekordu <End CKTP>

Przy odtwarzaniu **z unieważnieniem** system wykonuje:

- jeżeli czytając dziennik od końca napotka na wpis <End CKTP> to system unieważnia transakcje niezakończone te które rozpoczęły się po rekordzie <Start CKTP> oraz te będące na liście aktywnych transakcji
- Jeżeli czytając dziennik od końca napotka na wpis <Start CKTP> to jest on ignorowany tzn. szukamy pierwszego rekordu <End CKTP>

System NTFS zazwyczaj używa odtwarzania po awarii np. po nagłym zrestartowaniu systemu – czytany jest wtedy dziennik i przywracany spójny stan struktury systemu.
System nie odtwarza danych (tzn. zawartości plików) – byłoby to po pierwsze niezwykle kosztowne ze względu na pojemność dysku a po drugie przyjęto założenie, że to aplikacje same powinny dbać o ochronę danych i odtwarzanie np. tak jak robi to OpenOffice, w którym to prezentacja ta powstała :-)