Usporedba FAT32 i NTFS datotečnog sustava u Windows računalima

Draganjac, Katarina

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Preddiplomski stručni studij Elektrotehnika, smjer Informatika

USPOREDBA FAT32 I NTFS DATOTEČNIH SUSTAVA U WINDOWS RAČUNALIMA

Završni rad

Katarina Draganjac

Osijek, 2020.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1S: Obrazac za imenovanje Povjerenstva za završni ispit na preddiplomskom stručnom studiju

Osijek, 17.08.2020.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za završni ispit na preddiplomskom stručnom studiju

Ime i prezime studenta:	Katarina Draganjac
Studij, smjer:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI4537, 25.09.2019.
OIB studenta:	81550989168
Mentor:	Goran Bokun
Sumentor:	-
Sumentor iz tvrtke:	-
Predsjednik Povjerenstva:	Prof. dr. sc. Goran Martinović
Član Povjerenstva 1:	Goran Bokun
Član Povjerenstva 2:	Izv. prof. dr. sc. Ivica Lukić
Naslov završnog rada:	Usporedba FAT32 i NTFS datotečnih sustava u Windows računalima
Znanstvena grana rada:	**Procesno računarstvo (zn. polje računarstvo)**
Zadatak završnog rada	Ukratko objasniti datoteke i direktorije. Dati pregled datotečnih sustava općenito. Opisati FAT32 datotečni sustav. Opisati NTFS datotečni sustav. Dati usporedbu ta dva sustava.
Prijedlog ocjene pismenog dijela ispita (završnog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3. razina
Datum prijedloga ocjene mentora:	17.08.2020.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 07.09.2020.

Ime i prezime studenta:	Katarina Draganjac
Studij:	Preddiplomski stručni studij Elektrotehnika, smjer Informatika
Mat. br. studenta, godina upisa:	AI4537, 25.09.2019.
Turnitin podudaranje [%]:	6

Ovom izjavom izjavljujem da je rad pod nazivom **Usporedba FAT32 i NTFS datotečnih sustava u Windows računalima**

izrađen pod vodstvom mentora Goran Bokun

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.

Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Zadaća svakog računalnog programa je pohrana i dohvaćanje informacija, međutim tu se javlja nekoliko problema. Primjerice, kada se proces izvrši, informacije se gube. Za većinu aplikacija, kao što su baze podataka, informacije moraju ostati pohranjene mjesecima, čak i godinama te je neprihvatljiv njihov gubitak. Također, informacije ne bi smjele biti izgubljene niti kada dođe do iznenadnog prekida procesa.

Drugi problem u vezi pohrane informacija unutar adresnog prostora procesa je da tijekom izvršavanja procesa on može pohraniti samo ograničenu količinu informacija unutar svog adresnog prostora. Dakle, kapacitet pohrane ograničen je na veličinu adresnog prostora te je u današnje vrijeme ta veličina za većinu aplikacija premala.

Treći problem je što često nekoliko procesa treba pristupiti nekoj informaciji u isto vrijeme. Ako je određena informacija smještena u adresni prostor jednog procesa, samo taj proces može pristupiti toj informaciji. Taj problem može se riješiti tako da se informacija napravi neovisnom od bilo kojeg procesa.

Dakle, tri su osnovna zahtjeva za dugotrajnu pohranu podataka:

1. Mogućnost očuvanja informacija i nakon završetka procesa.
2. Mogućnost pohrane velike količine informacija.
3. Mogućnost pristupa nekoliko procesa informaciji istovremeno. [1]

Rješenje se može pronaći u magnetnim, optičkim ili SSD (*solid-state drive*) diskovima koji se predstavljaju kao linearni niz blokova fiksne veličine. Da bi se diskovi mogli koristiti za dugotrajnu pohranu podataka, potrebno je riješiti nekoliko pitanja, na primjer kako pronaći određenu informaciju ili kako znati koji je blok prazan.

Ovaj problem može se riješiti uvođenjem apstrakcije – datoteke (engl. *file*). Datoteka je logička jedinica informacije koju kreira proces. Disk obično sadrži tisuće datoteka neovisne jedna od druge, a procesi mogu čitati postojeće datoteke ili kreirati nove ukoliko je potrebno. Informacije pohranjene u datotekama su trajne, odnosno početak ili završetak procesa na njih nema utjecaja, a datoteka nestaje samo ukoliko je korisnik izbriše.

Datotekama upravlja operacijski sustav te o njemu ovisi kako su one strukturirane, kako im se pristupa, kako ih se imenuje, koristi i slično. Dio operacijskog sustava koji je odgovoran za upravljanje datotekama naziva se datotečni sustav (engl. *file system*). Datotečni sustav se brine, između ostalog, i o smještanju datoteka, očuvanju podataka, sigurnosti i oporavka u slučaju pogrešaka. Jednostavnije rečeno, njegova uloga je ispuniti sve zahtjeve za pohranu podataka koje su navedene ranije.

Svaki operacijski sustav ima svoje datotečne sustave, ali je moguće da pojedini operacijski sustav podržava rad s datotečnim sustavima drugih operacijskih sustava. Na primjer, Linux podržava rad s NTFS, FAT32, exFAT te HFS-om. Mac podržava FAT32 i exFAT, ali NTFS može samo čitati, a ext datotečne sustave ne podržava. Windows podržava ext datotečne sustave, a HFS može samo čitati. Bitno je napomenuti da je u nekim od navedenih primjera potrebno koristiti posebne softvere koji omogućuju rad s određenim datotečnim sustavima.

- Mac:
    - HFS+ (*Hierarchical File System*)
    - APFS (*Apple File System*)
- Linux:
    - Btrfs (*B-tree File System*)
    - XFS (*Extents File System*)
    - JFS (*Journaled File System*)
    - ext (*extended*): ext2, ext3, ext4
- Windows:
    - FAT (*File Allocation Table*): FAT12, FAT16, FAT32, FATX, exFAT
    - NTFS (*New Technology File System*)
    - ReFS (*Resillient File System*) [2]

FAT ima nekoliko inačica, a ovaj rad bavit će se najpoznatijom - FAT32. Nekada je FAT bio zadani datotečni sustav u Microsoft Windows računalima, ali kako se povećavala trajna memorija računala tako ga je s vremenom zamijenio NTFS. Međutim, FAT32 još je uvijek prisutan, između ostalog, na USB uređajima.

Zadatak ovog rada je usporediti FAT32 i NTFS datotečne sustave koji su sastavni dio Windows operacijskih sustava. U sljedećem poglavlju pobliže će biti pojašnjen pojam datoteka i direktorija koji su nužni za shvaćanje koncepta datotečnih sustava općenito. Zatim će FAT32 i NTFS biti pojedinačno detaljno objašnjeni, a na kraju će biti dana usporedba ova dva datotečna sustava.

2. DATOTEKE I DIREKTORIJI

Dvije su ključne apstrakcije operacijskog sustava: proces, tj. virtualizacija procesora i adresni prostor, tj. virtualizacija memorije. Dakle, ove dvije apstrakcije omogućuju programu da se izvršava kao da je izoliran od ostalih komponenata, odnosno kao da ima svoj procesor i svoju memoriju. S obzirom da je često potrebno i sačuvati podatke nastale radom nekog programa, logičan nastavak ovog koncepta je bilo dodavanje još jedne apstrakcije: trajna pohrana podataka.

Uređaji za trajnu pohranu podataka, na primjer tvrdi disk (HDD – *hard disk drive*) ili SSD, kao što i sam naziv govori trajno pohranjuju podatke te, za razliku od radne memorije čiji se spremnik briše nakon završetka procesa, održavaju podatke netaknutima. Dakle, operacijski sustav mora posebnu pažnju posvetiti takvim uređajima jer su podatci spremljeni na njima jako bitni krajnjim korisnicima. Tijekom vremena su se zbog toga razvile dvije apstrakcije, odnosno virtualizacije pohrane podataka: datoteka i direktorij.

2.1. Datoteka

Datoteka je u svojoj osnovi linearni slijed bajtova, a omogućuje pohranu podataka te njihovo kasnije dohvaćanje. Naravno, to se odvija na način da korisnik ne zna detalje o tome kako se podatci spremaju i gdje, odnosno kako diskovi zapravo funkcioniraju.

Bitno je napomenuti da postoje različite vrste datoteka od kojih je samo jedna namijenjena korisniku, odnosno za pohranu korisničkih informacija, a to su tzv. regularne ili standardne datoteke (engl. *regular files*). Ostale vrste uključuju *character special* datoteke za I/O (*Input/Output*) uređaje i *block special* datoteke za diskove. Ovdje će biti riječi o regularnim datotekama.

Jedna od glavnih karakteristika datoteke je njezino imenovanje. Pravila imenovanja ovise o samom operacijskom sustavu, a za MS Windows računala vrijedi da se ne dopuštaju znakovi /, \, :, *, ?, “, <, > i | u nazivu datoteke, a sve datoteke istog tipa, odnosno ekstenzije (.pdf, .docx i sl.) ne mogu imati isti naziv, osim ako se nalaze u različitim direktorijima. Ekstenzija datoteke zapravo predstavlja tip datoteke, odnosno govori je li datoteka spremljena kao npr. *Word* dokument ili kao slika i ta informacija se može pronaći u svojstvima (engl. *properties*) pojedine datoteke.

Velika i mala slova među datotekama se ne razlikuju (*case-insensitive*), ali se čuvaju (*case-preserving*). To znači da ako se spremi datoteka Test.txt, Windows će ju tako i sačuvati. Međutim, ako bi u tražilici bilo upisano test.txt, Windows bi prepoznao da se zapravo radi o Test.txt. Također, kao što je prethodno rečeno, nije moguće imati dvije datoteke istog imena, a različitih veličina slova u istom direktoriju.

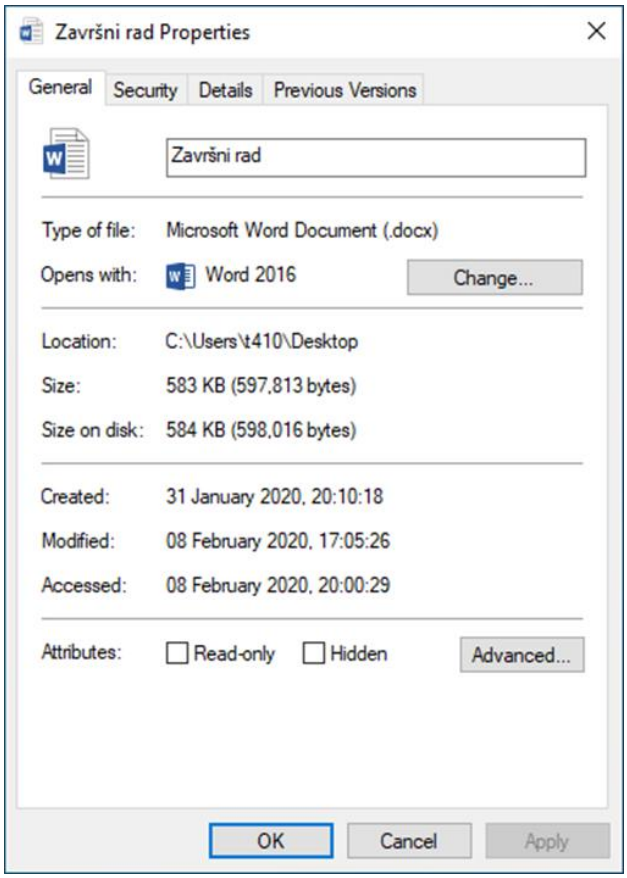
Maksimalan broj znakova u starijim verzijama Windows datotečnih sustava bio je tzv. *8.3 alias*, odnosno 8 znakova za naziv datoteke, a tri znaka za ekstenziju, dok je u novijim sustavima broj znakova za naziv datoteke u pravilu 260. Međutim, to ovisi o tome i gdje se datoteka nalazi jer se u tih 260 znakova računa i putanja (engl. *path*) do te datoteke.

Na primjer, ako se datoteka nalazi na radnoj površini, njezina putanja može izgledati ovako: *C:\Users\UserName\Desktop\FileName.ext*. Dakle, naziv datoteke u ovom slučaju mogao je biti 260 - 31, koliko zauzima putanja i ekstenzija datoteke te *null* znak. [3]

Druga važna karakteristika datoteke su njezini meta podatci (engl. *metadata*). Svaka datoteka ima svoje ime i podatke, ali osim toga, operacijski sustav veže još neke informacije uz njih, kao što su datum i vrijeme kada je datoteka kreirana, veličina datoteke i sl. Te dodatne informacije se nazivaju meta podatci i mogu se pronaći klikom na opciju Svojstva (engl. *Properties*) i prikazani su na slici 2.1. Kartica *General* daje uvijek iste informacije, tj. ista je za sve tipove datoteka.

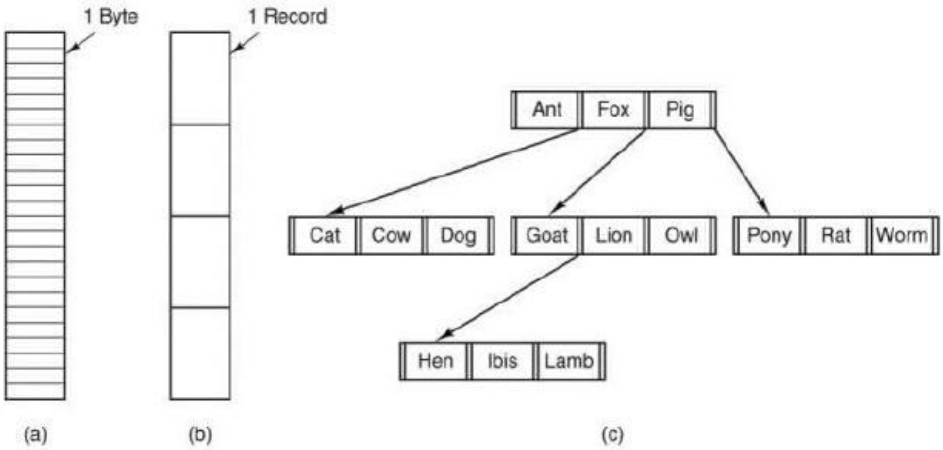
Kartica *Details* ovisi o tipu datoteke, pa će tako za *Word* dokument dati informacije o broju stranica, riječi, vremenu pisanja i sl., dok će za sliku dati informacije o njezinim dimenzijama i uređaju kojim je snimljena. [4]

Posebna pozornost se obraća na meta podatak kojeg također možemo vidjeti na slici 2.1., a to su atributi (engl. *attributes*). Atributi su posebna vrsta meta podataka koji definiraju ponašanje datoteke. Četiri su osnovna atributa: *Read-only* (R) označava da se datoteka može samo čitati, tj. ne može se izmjenjivati, *Hidden* (H) govori da se datoteka ne prikazuje na listi datoteka, *Archive* (A) upozorava da nije napravljen *backup* datoteke i *System* (S) predstavlja datoteku OS-a. [5]



Slika 2.1. *Meta podatci datoteke (Windows 10)*

Još jedna karakteristika datoteke zapravo je nešto što se odvija u pozadini i nije vidljivo krajnjem korisniku, a to je strukura datoteke. Postoje tri vrste struktura datoteka: *Byte sequence*, *Record sequence* i *Tree* te ih se može vidjeti na slici 2.2.



Slika 2.2. *Vrste struktura datoteka*

Kod niza bajtova (engl. *byte sequence*) operacijski sustav ne zna što je u datoteci i sve što vidi su bajtovi, tako da se korisnički programi brinu oko samog prikaza korisniku. Ovakva struktura pruža potpunu fleksibilnost jer se operacijski sustav ne miješa u podatke datoteka niti u njihove nazive. Dakako, svaka datoteka ima svoje zaglavlje koje sadrži podatke o adresiranju ostatka datoteke. Ovakav pristup koriste Windows operacijski sustavi.

Kod niza zapisa (engl. *record sequence*) datoteka je niz zapisa fiksne veličine. Ključna ideja je bila da program dohvaća jedan zapis ili sprema podatke u drugi zapis. Međutim, ova vrsta strukture je imala primjenu davno kada su se koristile bušene kartice (engl. *punched cards*) te je to bilo najučinkovitije riješenje, a danas se uglavnom ne koristi.

Zadnja vrsta je struktura stabla (engl. *tree*) koja predstavlja stablo zapisa koji ne moraju biti jednake duljine, a svaki sadrži ključno polje (engl. *key field*) koje se nalazi na fiksnoj poziciji u zapisu. Ključno polje omogućuje brzo dohvaćanje pojedinog zapisa. Ova struktura se koristi samo na *mainframe* računalima.

Posljednja karakteristika datoteka koja će biti spomenuta je pristup datotekama. U početku su operacijski sustavi pružali mogućnost samo sekvencijalnog pristupa (engl. *sequential access*), odnosno proces je morao čitati podatke redom od početka do kraja, bez preskakanja što je bilo prihvatljivo za magnetne trake. Međutim, pojava diskova omogućila je nasumičan pristup (engl. *random access*) podatcima. Dakle, podatci se ne moraju čitati redom nego se bilo kojem podatku na disku može pristupiti posebno, pod pretpostavkom da se zna gdje se taj podatak nalazi, što je naravno omogućilo puno brži pristup datotekama. [1]

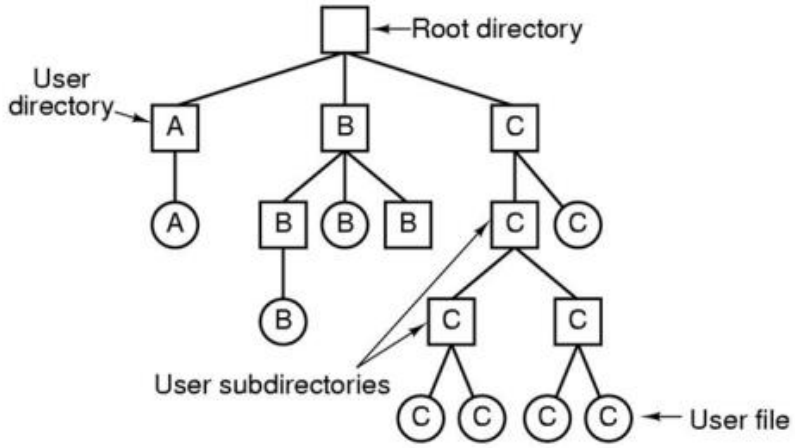
2.2. Direktorij

Nastankom koncepta datoteka, morao je nastati i koncept upravljanja njima te se tako dolazi do direktorija (engl. *directory*). Direktorij je bitan dio samog datotečnog sustava, odnosno on određuje kako će datoteke biti organizirane na računalu, tj. kako će im se pristupati. Ponekad se zapravo smatra samo posebnom vrstom datoteke, a koja sadrži informacije o strukturi datotečnog sustava.

Strukture direktorija su sljedeće: jednorazinski (engl. *single-level)*, dvorazinski (engl. *two-level*) te stablasti (engl. *tree-structured, hierarchical*) direktorij. Jednorazinski je najjednostavnija struktura direktorija te su u takvoj strukturi sve datoteke smještene unutar jednog direktorija.

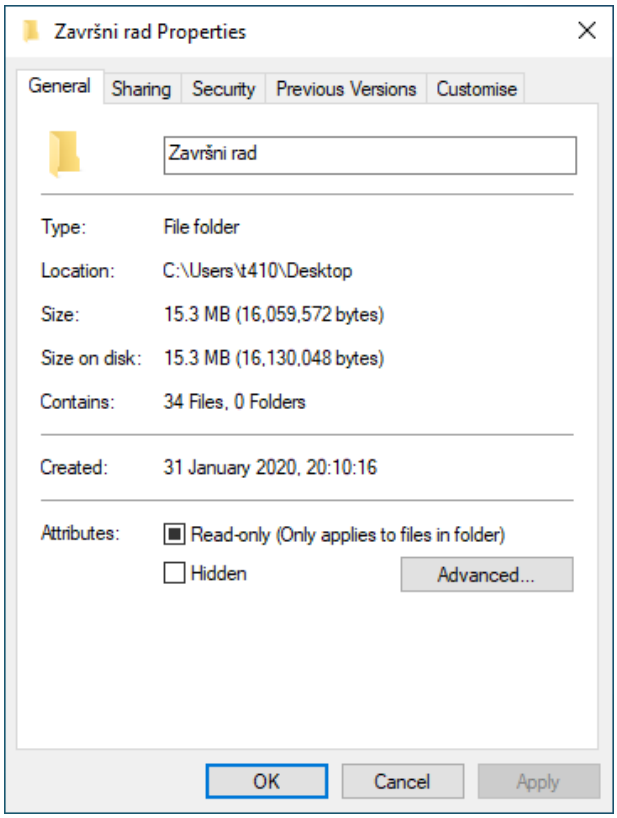
Jednorazinska struktura ima ograničenja kada dođe do povećanog broja datoteka (manjak organizacije i imenovanje), pa je kao rješenje nastao dvorazinski koji je uveo direktorije korisnika. Dakle, svaki korisnik je mogao imati jedan direktorij u kojemu su bile sve njegove datoteke.

Stablasta struktura najuobičajenija je struktura direktorija, odnosno struktura koju koriste gotovo svi moderni operacijski sustavi, a prikazana je na slici 2.3. Svaki korisnik ima mogućnost kreirati svoje poddirektorije u kojima se nalaze datoteke, a glavni direktorij, od kojeg sve započinje, naziva se *root* (korijenski) direktorij. Pronalazak pojedinih datoteka vrlo je jednostavan jer su sve datoteke smisleno organizirane, međutim nedostatak je ukoliko postoji puno direktorija unutar direktorija, tada je potrebno više vremena da se dođe do neke određene datoteke. [6]



Slika 2.3. *Hijerarhijska struktura*

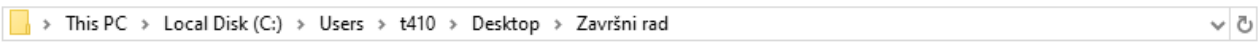
Kao što je već spomenuto, operacijski sustav smatra direktorije samo posebnim vrstama datoteke pa tako neke karakteristike koje vrijede za datoteke vrijede i za direktorije, npr. pravila imenovanja te atributi. Direktoriji također imaju meta podatke koji se mogu pronaći klikom na opciju *Properties* kao i kod datoteka. Na slici 2.4. se može vidjeti kako se svojstva direktorija malo razlikuju u odnosu na svojstva datoteka. To je zbog toga što se direktorij ne može izravno izmjenjivati nego samo njegove datoteke pa on nema podatke *Modified* i *Accessed*, ali zato ima podatak o broju datoteka i poddirektorija koje sadrži.



Slika 2.4. *Svojstva direktorija (Windows 10)*

S obzirom da su direktoriji i datoteke organizirane hijerarhijski mora postojati način za određivanje položaja datoteka. Zbog toga svaka datoteka i direktorij u sustavu imaju svoju putanju (engl. *path*), a oznaka „\“ odvaja nazive datoteka i direktorija u putanji. Dva su tipa putanja: apsolutna i relativna. Apsolutna putanja specificira položaj datoteke ili direktorija u odnosu na cijeli datotečni sustav te uvijek započinje s *root* direktorijem, dok relativna putanja određuje položaj datoteke ili direktorija u odnosu na trenutni direktorij (engl. *working/current directory*), tako da je moguće jednostavno pristupiti datoteci ili direktoriju u radnom direktoriju samo navođenjem njezina imena. Ako je trenutni direktorij *UserFolder*, a datoteka se nalazi u njegovom poddirektoriju koji se naziva *UserSubFolder*, primjer relativne putanje je: *UserSubFolder/UserFile.ext*. Na slici 2.5. može se vidjeti primjer apsolutne putanje prikazan u komandnoj liniji (engl. *command prompt)*, dok se na slici 2.6. može vidjeti prikaz apsolutne putanje koju pruža operacijski sustav kroz svoje grafičko sučelje. [7]

Slika 2.5. *Primjer putanje u komandnoj liniji*

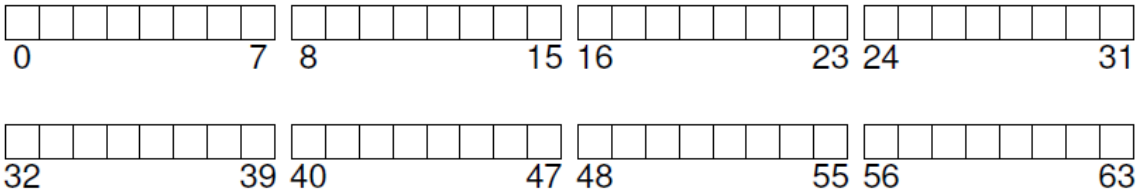


Slika 2.6. *Primjer putanje u Windows 10 operacijskom sustavu*

3. DATOTEČNI SUSTAVI

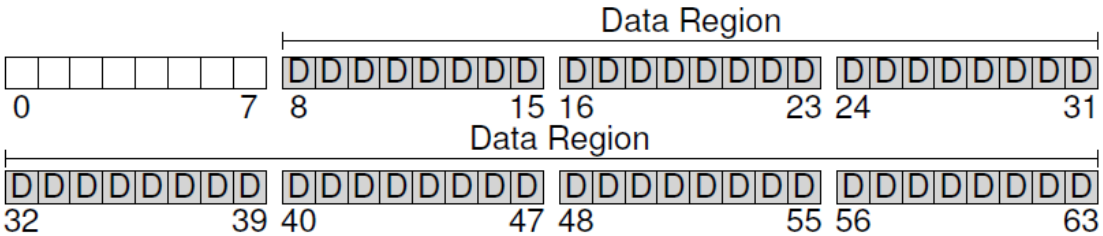
Upravljanje trajnom memorijom vrlo je bitan zadatak računala, a operacijski sustav pruža tu mogućnost putem datotečnog sustava (engl. *file system*). Dakle, datotečni sustav je način upravljanja podatcima na trajnoj memoriji računala, odnosno omogućuje operacijskom sustavu da razlikuje različite vrste datoteka (standardne, specijalne i sl.), te omogućuje operacije nad datotekama, kao što su stvaranje, brisanje, otvaranje, zatvaranje, čitanje i slično. Kod promatranja datotečnog sustava bitne su dvije stavke: struktura podataka te načini pristupa.

Ove stavke bit će objašnjene na primjeru jednostavnog datotečnog sustava. Prije svega potrebno je zamisliti općenitu organizaciju strukture podataka, odnosno podijeliti disk na blokove (engl. *blocks*) jednakih veličina, na primjer 4KB. Dakle, disk bi izgledao kao niz blokova, svaki veličine 4KB, označeni s brojevima od 0 do N-1, gdje je N = 64 u ovom slučaju. Takav prikaz se može vidjeti na slici 3.1.



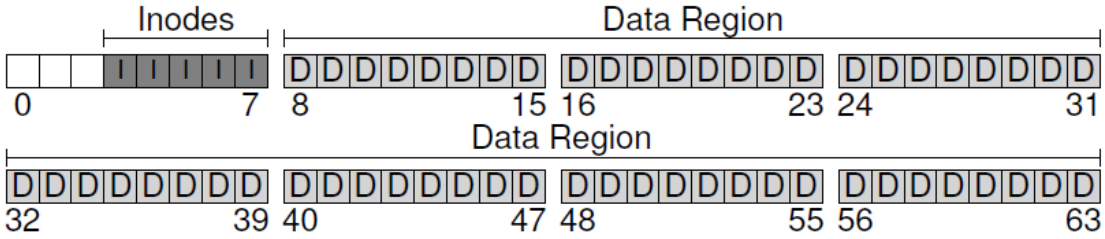
Slika 3.1. *Podjela diska na blokove*

Sljedeće što se mora uzeti u obzir je što se sve mora spremiti u te blokove kako bi se stvorio datotečni sustav. Prva stvar su naravno korisnički podatci (engl. *user data*) koji i inače čine najveći dio datotečnog sustava. Dio diska koji sadrži korisničke podatke bit će nazvan područjem podataka (engl. *data region*), a to se može vidjeti na slici 3.2.



Slika 3.2. *Područje podataka*

U prošlom poglavlju spomenuto je da datotečni sustav mora pohraniti i meta podatke svake datoteke, dakle informacije kao što su veličina datoteke, datum kreiranja, prava pristupa i sl. Kako bi datotečni sustav uspješno zapamtio te podatke, uvedena je struktura *inode* (*index-node,* indeks-čvor), a koja će u ovom primjeru predstavljati općeniti naziv za strukturu koja ima za zadaću pamtiti meta podatke pojedinih datoteka. U datotečnim sustavima za Linux računala ova struktura se uistinu i zove *inode*, međutim, većina današnjih sustava ima ovakvu strukturu, ali uglavnom drugog naziva. Dakle, mora se osigurati prostor na disku i za *inode* što se može vidjeti na slici 3.3.

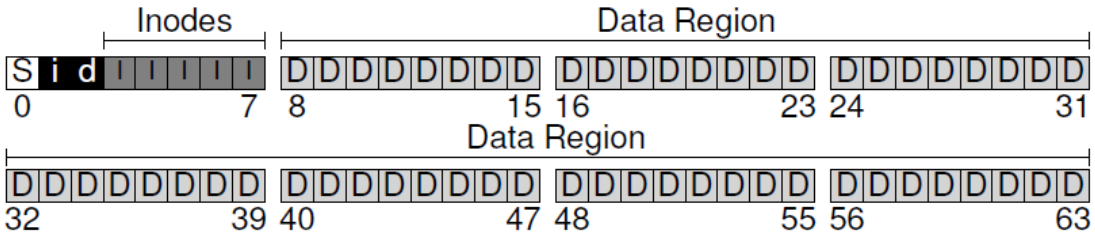


Slika 3.3. *Inode*

Zatim je potreban način dohvaćanja informacija o tome jesu li područja *inode-a* ili podataka slobodna ili zauzeta, a za taj problem postoji nekoliko rješenja. Jedan od njih je *free list* u kojemu postoji glavni blok s pokazivačem na prvi slobodni blok, a taj blok pokazuje na sljedeći slobodni blok i tako se formira lista slobodnih blokova. Jednostavnije rješenje je struktura poznata kao *bitmap,* a potreban je jedan za *inode* područje (engl. *inode bitmap*) i jedan za područje podataka (engl. *data bitmap*), odnosno koristi se jedan bit za provjeru je li odgovarajući blok slobodan (0) ili zauzet (1). Dakle, kada se kreira datoteka mora se alocirati *inode* za tu datoteku, a datotečni sustav će pretražiti *bitmap* kako bi pronašao slobodan *inode,* dodijelio ga datoteci i označio ga zauzetom (1).

Na kraju se mora još popuniti jedan preostali blok, a taj blok namijenjen je tzv. superbloku. Superblok sadrži informacije o pojedinom datotečnom sustavu, kao na primjer koliko je *inode* i podatkovnih blokova, koji blok označava početak *inode-a* i slično. Dakle, pri pokretanju operacijskog sustava, prvo što će pročitati je superblok kako bi mogao inicijalizirati razne parametre. [8]

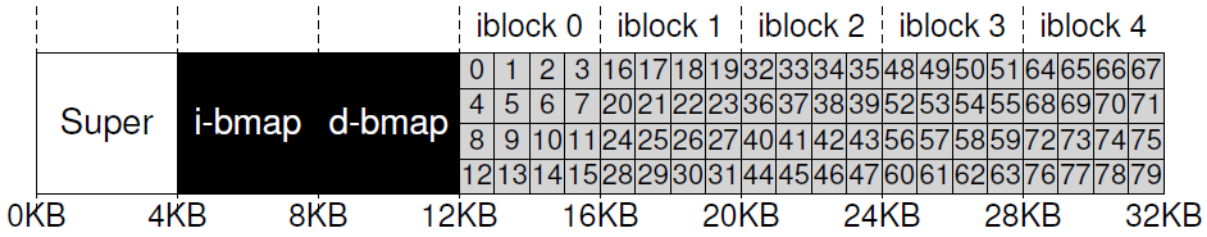
Bitmap i superblok se može vidjeti na slici 3.4.



Slika 3.4. *Inode bitmap (i), data bitmap (d) i superblok (S)*

3.1. Organizacija datoteka

Jedna od najbitnijih struktura vezana uz datoteke je već spomenuti *inode* (*index node*). Unutar svakog *inode-a* sadržane su sve potrebne informacije o datoteci, a svaki *inode* ima oznaku broja preko kojeg mu se može pristupiti (i-broj, engl. *i-number*), a ujedno je to i niskorazinski (engl. *low-level*) naziv datoteke, odnosno naziv s kojim radi sustav. Pomoću i-broja može se izračunati gdje se nalazi odgovarajući *inode*. Na primjer, ako se uzme podjela diska koja je spomenuta u prethodnom poglavlju i pobliže se pogleda *inode* dio, dobiva se tablica veličine 20KB, odnosno pet blokova veličine 4KB koji ukupno čine 80 *inode-a*, što se može vidjeti na slici 3.5.

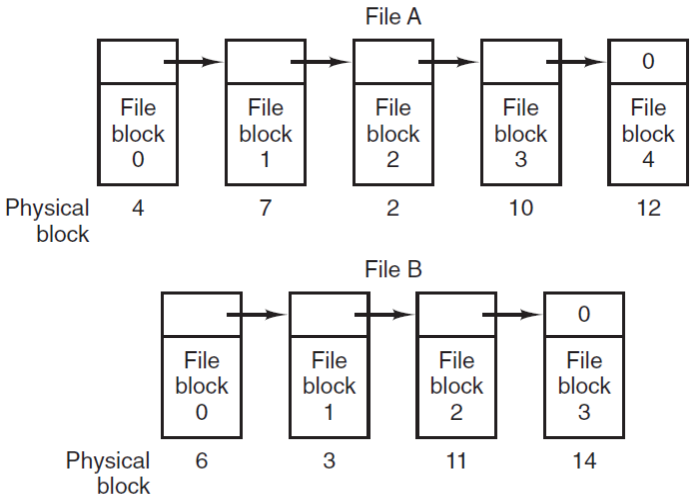


Slika 3.5. *Tablica inode*

Sljedeće što je preostalo odrediti je način na koji će *inode* pokazivati na određene podatkovne blokove. Najjednostavniji pristup je putem direktnih pokazivača (engl. *direct pointers*) unutar *inode-a*, gdje svaki pokazivač pokazuje na jedan blok diska koji pripada datoteci. Takav pristup je ograničen ukoliko se radi o većim datotekama. Kako bi se riješio problem većih datoteka uvedeni su indirektni pokazivači (engl. *indirect pointers*). Dakle, umjesto da pokazivač pokazuje na blok koji sadrži podatke, on pokazuje na blok koji sadrži još nekoliko pokazivača, od kojih svaki pokazuje na podatkovni blok. Ukoliko se žele podržati još veće datoteke, može se uvesti dodatni pokazivač. Tada se dobije dvostruko indirektni pokazivač (engl. *double indirect pointer*) koji pokazuje na blok indirektnih pokazivača koji pokazuju na blok direktnih pokazivača koji

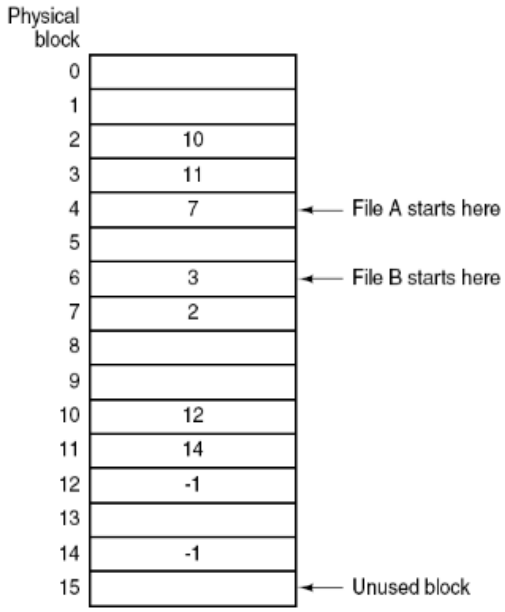
pokazuju na podatkovne blokove. Postoji mogućnost uvođenja nekoliko dodatnih pokazivača pa se tako može govoriti i o višerazinskom (engl. *multi-level index*) pristupu pokazivanja na podatkovne blokove. Ovakav način pristupa prisutan je kod ext2 i ext3 datotečnih sustava za Linux računala. [8]

Sljedeći način pristupa podatkovnim blokovima o kojem će se govoriti su povezane liste (engl. *linked lists*). Dakle, unutar *inode-a*, umjesto nekoliko pokazivača, potreban je samo jedan koji pokazuje na početak prvog bloka datoteke. Na kraju prvog bloka postavlja se pokazivač koji pokazuje na sljedeći blok i tako redom. Primjer se može vidjeti na slici 3.6.



Slika 3.6. *Povezana lista*

Nedostatak ovakvog pristupa je što je vremenski vrlo zahtjevan pronalazak nasumičnih blokova ili primjerice pristup zadnjem bloku pa je kasnije, umjesto samih pokazivača unutar svakog bloka, uvedena tablica unutar memorije koja sadrži informacije o poveznicama. Tako je omogućen puno brži pronalazak određenih blokova s obzirom da se prvo pretraži sama tablica kako bi se pronašla lokacija određenog bloka u memoriji, a tek mu se potom pristupa na disku. Na slici 3.7. se može vidjeti da datoteka A zauzima 4, 7, 2, 10 i 12 blok, dok datoteka B zauzima 6, 3, 11 i 14 blok, a u oba slučaja vrijednost -1 označava kraj datoteke.



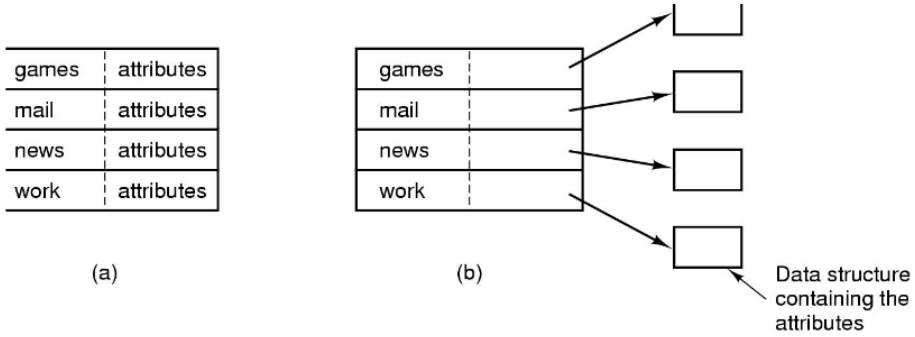
Slika 3.7. *Povezane liste pomoću tablice*

Ovakav način pristupa blokovima datoteke poznat je kao *file allocation table*, odnosno temelj je FAT datotečnog sustava o kojem će biti riječi u četvrtom poglavlju. [1]

3.2. Organizacija direktorija

Direktoriji općenito imaju jednostavnu organizaciju, odnosno direktorij zapravo sadrži listu naziva datoteka i informacije vezane uz njih. Dakle, za svaku datoteku ili direktorij unutar nekog direktorija postoje potrebne informacije za pronalazak njihovih podatkovnih blokova. Ovisno o načinu implementacije datoteka, razlikuje se i sami sadržaj informacije. U svakom slučaju, zadaća direktorija je ASCII naziv datoteke ili direktorija povezati s odgovarajućom informacijom koja se nalazi u listi kako bi se pronašao određeni podatak.

Tako se dolazi do pitanja gdje spremiti te potrebne informacije, odnosno atribute. Prva mogućnost je spremiti ih direktno u zapis direktorija, dakle jednostavno rečeno direktorij se tada sastoji od liste zapisa fiksne veličine, po jedan za svaku datoteku u kojoj se nalazi ime datoteke te njeni atributi i adrese podatkovnih blokova. To se može vidjeti na slici 3.8./a. Kod sustava koji koriste *inode* postoji druga mogućnost koja se može vidjeti na slici 3.8./b. U tom slučaju informacije spremamo u *inode* te tada direktorij sadrži samo naziv i broj *inode-a*. Ovaj način ima više prednosti u odnosu na prethodni, primjerice s obzirom da se u prvom slučaju mora odrediti fiksna veličina zapisa, onda će u nekim slučajevima nastati neiskorišteni prostor jer neće svaki zapis biti iste veličine.



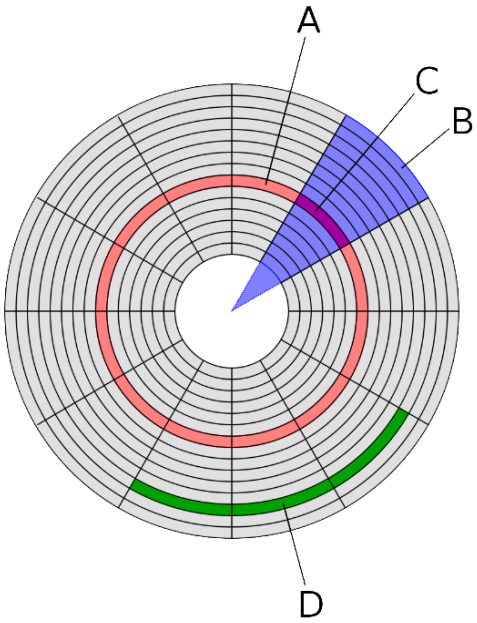
Slika 3.8. *Pohrana informacija: a) unutar direktorija b) izvan direktorija*

Sljedeće pitanje koje se postavlja je kako su sami direktoriji spremljeni. U većini slučaja datotečni sustavi se prema direktorijima odnose kao prema posebnoj vrsti datoteke. Prema tome, svaki direktorij ima oznaku u *inode* tablici (ili nekakvoj sličnoj strukturi) koja sadrži informaciju da se radi o direktoriju, a ne običnoj datoteci. Kao i kod datoteka, *inode* pokazuje na određeni podatkovni blok vezan uz taj direktorij. [8]

4. FAT32 I NTFS DATOTEČNI SUSTAVI

Do sada je bilo rečeno kako je neophodno bilo uvesti strukturu koja će omogućiti organizaciju trajne pohrane podataka, a ta struktura se naziva datotečni sustav. Temelj datotečnih sustava su datoteke koje predstavljaju podatke te direktoriji koji služe za dodatnu organizaciju. Datotečni sustav određuje kako će izgledati struktura datoteka i direktorija na disku, odnosno kako će se točno spremati na disk, kako će im se pristupati i slično. U prethodnim poglavljima ovi koncepti su bili objašnjeni općenito, a u ovom poglavlju će to biti prikazano na konkretnim primjerima dva datotečna sustava Windows računala – FAT32 i NTFS. Međutim, kako bi se u potpunosti mogao shvatiti princip rada nekog datotečnog sustava potrebno je ukratko vidjeti kako izgleda tvrdi disk, koji je među najpopularnijim načinima trajne pohrane podataka.

Slika 4.1. *Unutrašnjost tvrdog diska*

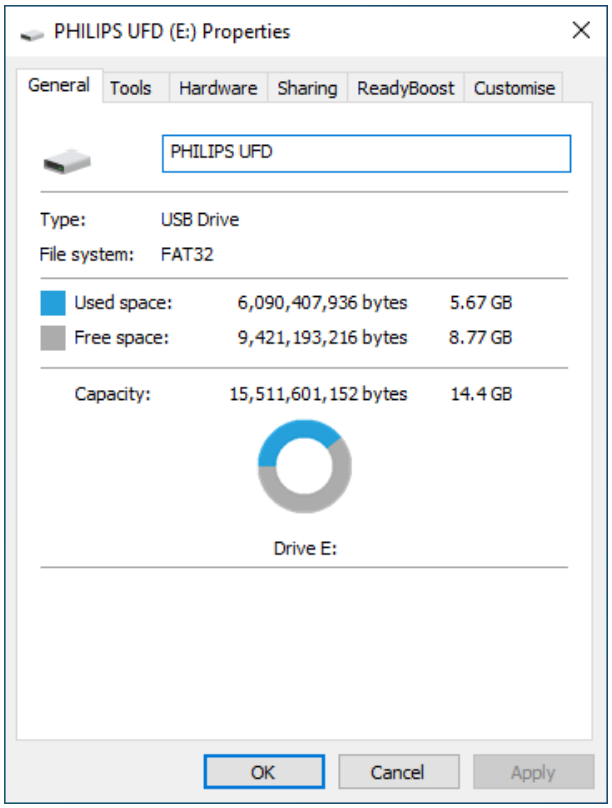


Slika 4.2. *Dijelovi ploče*

Na slici 4.1. može se vidjeti unutrašnjost tvrdog diska, odnosno njegove najbitnije dijelove – magnetna ploča, osovina, magnetna glava, pokretač te ruka pokretača. Ono što je bitno je organizacija podataka na magnetnoj ploči, odnosno na pojedinom disku. Svaki disk se prilikom formatiranja može podijeliti na particije, a organizacija podataka na particiji se može vidjeti na slici 4.2. Logički dijelovi ploče su: traka (A), sektor (B), sektor trake (C) i *cluster* (D). Dakle, disk je podijeljen na koncentrične krugove, tj. trake kojih može biti i na tisuće. Svaka traka podijeljena je na sektore koji predstavljaju najmanju jedinicu za pohranu podataka na disku te svaka traka ima jednak broj sektora. *Cluster* se sastoji od određenog broja sektora. [9][10]

4.1. FAT32

FAT (*File Allocation Table*) datotečni sustav razvijen je 1977. godine kao jednostavan datotečni sustav pogodan za diskete (engl. *floppy disk*). Međutim, s vremenom je poboljšan te se koristio dva desetljeća, od MS-DOS do Windows 9x operacijskih sustava. S razvojem naprednijih računala kao i operacijskih sustava, napravljeni su i složeniji datotečni sustavi pa tako FAT više nije zadani sustav MS Windows operacijskih sustava. U današnje vrijeme koristi se na USB i *flash* memoriji te memorijskim karticama, a najviše iz razloga što je zbog svoje jednostavnosti kompatibilan sa svim operacijskim sustavima. Najpoznatija tri tipa FAT datotečnih sustava su FAT12, FAT16 te FAT32, ali postoje još i verzije exFAT, FATX te FAT+. [11]



Slika 4.3. *Primjer USB diska s FAT32 datotečnim sustavom (Windows 10)*

Ime ovog datotečnog sustava dolazi od toga što koristi tablicu koja se statički alocira prilikom formatiranja. Tablica sadrži zapis za svaki *cluster*, tj. određeni broj sektora diska, a svaki zapis sadrži ili broj sljedećeg *clustera*, oznaku kraja datoteke ili neiskorištenog prostora diska. Osnovna razlika između FAT tipova, kao i razlog broja u njihovom imenu, je veličina zapisa u FAT tablici, u bitovima. Dakle, FAT32 koristi 32 bita za svaki zapis clustera u tablici. [12]

FAT32 datotečni sustav prvi puta pojavljuje se 1996. godine na Windowsu 95, a sastoji se od tri područja (engl. *regions*):

- Rezervirano područje (engl. *Reserved region*)
- FAT područje (engl. *FAT region*)
- Podatkovno područje (engl. *Data region*) [13][14]

FAT datotečni sustav		
Rezervirano područje	**FAT područje**	**Podatkovno područje**

REZERVIRANO PODRUČJE

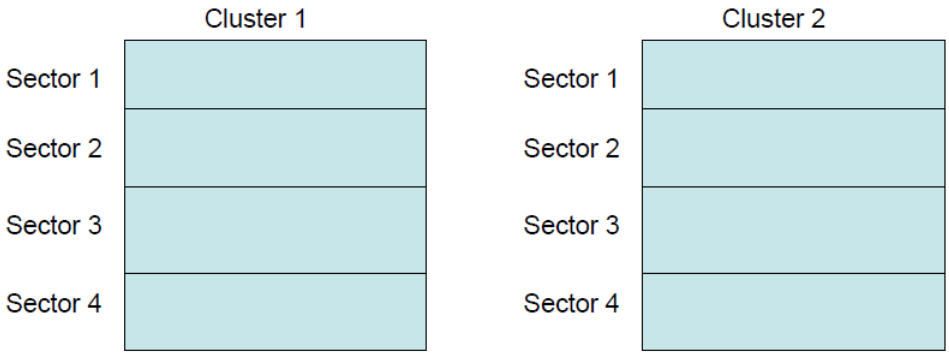
Prvi rezervirani sektor (sektor 0) je *Boot Sector*, poznat kao i *Volume Boot Record* – VBR. Najbitnija sastavnica ovog sektora je *BIOS Parameter Block* – BPB, a koji pruža sve temeljne informacije o FAT32 datotečnom sustavu. Te informacije se mogu vidjeti u tablici 4.1. Dakle, može se zaključiti da svaki sektor uvijek ima 512 bajtova (potencija broja 2 - 2^9), a broj sektora koji čine jedan *cluster* također uvijek mora biti potencija broja 2 zato što se sektori smještaju u 8-bitna polja. Uvijek postoje dvije FAT tablice, jedna glavna te druga rezervna. Ono što je specifično u ovom području za FAT32 je da sadrži i *File System Information Sector* (sektor 1) te *Backup Boot Sector* (sektor 6) koji se nisu nalazili u prethodnim verzijama FAT-a. *File System Information Sector* je uveden kako bi ubrzao vrijeme pojedinih operacija, primjerice dohvaćanje količine slobodnog prostora. *Backup Boot Sector* je koristan ako dođe do gubitka ili oštećenja podataka u *Boot sectoru* pa te podatke možemo jednostavno vratiti njihovim dohvaćanjem iz backupa. [13][15]

Polje	Microsoft naziv	*Offset*	Veličina	Vrijednost
Bajtovi po sektoru	BPB_BytsPerSec	0x0B	16 bita	Uvijek 512 bajtova
Sektori po clusteru	BPB_SecPerClus	0x0D	8 bita	1,2,4,8,16,32,64,128
Rezervirani sektori	BPB_RsvdSecCnt	0x0E	16 bita	Obično 0x20
Broj FAT tablica	BPB_NumFATs	0x10	8 bita	Uvijek 2
Sektori po FAT-u	BPB_FATSz32	0x24	32 bita	Ovisi o veličini diska
Početni cluster *root* direktorija	BPB_RootClus	0x2C	32 bita	Obično 0x00000002
Signature	(nema)	0x1FE	16 bita	Uvijek 0xAA55

Tablica 4.1. *Osnovne informacije FAT32 datotečnog sustava*

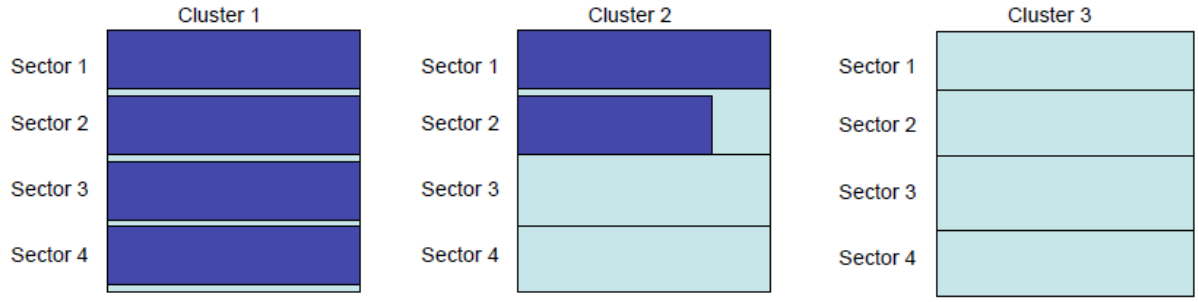
FAT I PODATKOVNO PODRUČJE

Kao što je ranije rečeno, kod FAT32 sustava disk, odnosno particija diska je podijeljena na podatkovne blokove jednakih veličina – clustere, koji se zapravo sastoje od određenog broja sektora, što se može vidjeti na slici 4.4.



Slika 4.4. *Podjela diska na clustere*

Prilikom pohrane datoteke na disk alocira se određeni broj clustera koji su joj potrebni. Kao što se može vidjeti na slici 4.5., datoteke uglavnom ne mogu popuniti svaki cluster do kraja te tada dolazi do neiskorištenog prostora koji se naziva *slack*. [14]



Slika 4.5. *Primjer datoteke koja zauzima dva clustera*

Veličina clustera ovisi o veličini particije diska, a zadane veličine clustera za FAT32 mogu se vidjeti u tablici 4.2. S obzirom na vrijeme kada je nastao, FAT32 optimiziran je za male particije diska. Ranije je maksimalna veličina particije u teoriji bila 2TB, ali od Windowsa 2000 nisu podržane particije veće od 32GB. Razlog tome je što veličina clustera postaje prevelika kako se povećava veličina particije što prije nije predstavljalo problem jer particije svakako nisu bile velike. Međutim, kako današnja računala imaju gotovo uvijek i puno više od 32GB memorije, veličina clustera postala je velik problem jer se stvaralo previše neiskorištenog prostora (*slacka*).

Veličina particije	> MS Windows 2000
7 MB – 16 MB	Nije podržano
16 MB – 32 MB	Nije podržano
32 MB – 64 MB	512 B
64 MB – 128 MB	1 KB
128 MB – 256 MB	2 KB
256 MB – 8 GB	4 KB
8 GB – 16 GB	8 KB
16 GB – 32 GB	16 KB
32 GB – 2 TB	Nije podržano
> 2 TB	Nije podržano

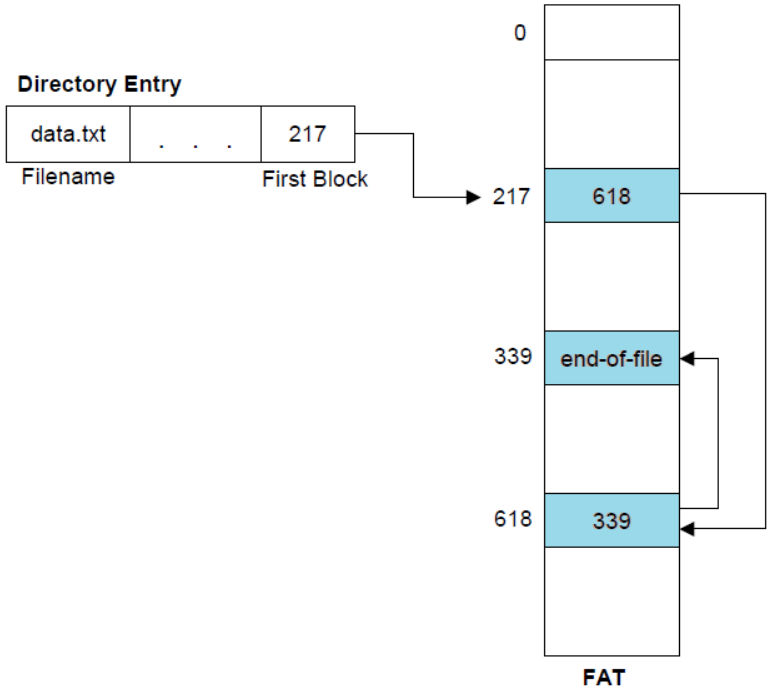
Tablica 4.2. *Zadane veličine clustera u FAT32 datotečnom sustavu* [16]

Ono što se postavlja kao pitanje je kako znati gdje se nalazi koja datoteka, odnosno koji cluster pripada kojoj datoteci. Za to nam služi FAT tablica koja je temelj FAT32 datotečnog sustava. Ona se nalazi na određenoj poziciji na disku, točnije u prvom sektoru (sektor 0) te ju čine polja jednakih veličina u kojima se nalaze zapisi. Broj zapisa odgovara broju clustera na disku, a sam zapis može predstavljati: slobodan cluster, kraj datoteke (*end-of-file*), oštećeni cluster, rezervirani cluster te sljedeći cluster u lancu. FAT datotečni sustavi uvijek imaju dvije FAT tablice – primarnu te kopiju (*backup*) iste.

Dakle, FAT je u svojoj osnovi povezana lista, o kojoj je bilo riječi u prošlom poglavlju. Ukratko, povezana lista koja je ostvarena pomoću tablice znači da tablica sadrži sve zapise o tome gdje se nalazi koji podatak na disku pa se tako ne mora pretraživati cijeli disk da bi se došlo do traženog podatka, nego se samo pretraži tablica, a zatim se pristupa samom podatku na disku.

U trećem poglavlju bilo je riječi o datotečnim sustavima općenito te je tada spomenuta struktura *inode* koja sadrži meta podatke pojedinih datoteka kao i informaciju o tome gdje se one nalaze u memoriji. U FAT32 datotečnom sustavu ovu ulogu ima struktura direktorija. Direktorij sadrži zapise o nazivima datotekama i svim informacijama vezanih uz njih, kao što su meta podatci, atributi te najbitnije - pozicija prvog clustera u kojem se datoteka nalazi. Glavni direktorij od kojeg sve započinje naziva se *Root Directory*, a kao što je rečeno ranije, u rezerviranom području se nalazi informacija o poziciji njegovog prvog clustera. [17]

Dakle, pristup nekoj datoteci započinje od direktorija u kojem se nalazi zapis o imenu tražene datoteke i poziciji njezinog prvog clustera. FAT ima zapise o pozicijama svakog sljedećeg clustera neke datoteke pa sve do zadnjeg koji sadrži oznaku kraja datoteke (slika 4.6.).



Slika 4.6. *Pristup clusterima* [18]

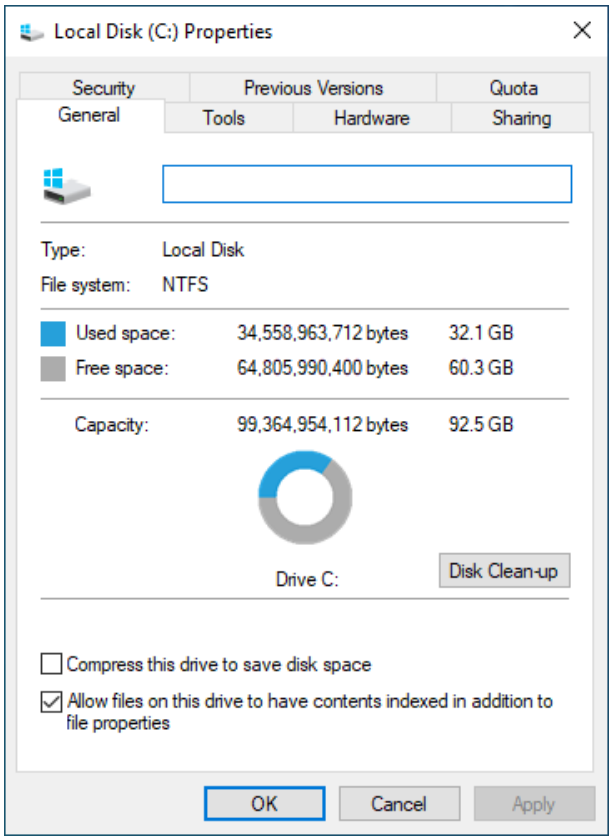
Kao što je već rečeno, svaki cluster ima svoju oznaku u FAT tablici, a jedna od tih oznaka je i da se radi o neiskorištenom, odnosno praznom clusteru. To nam omogućuje pohranu novih podataka, a za koju postoji nekoliko načina izbora clustera:

- *First Available* - kreće otpočetka sustava i podatak sprema u prvi slobodni cluster,
- *Best Fit* – traži slobodne susjedne clustere koji bi odgovarali veličini datoteke,
- *Next Available* - pretraga slobodnog clustera kreće od zadnje alociranog clustera.

Prilikom brisanja određene datoteke, njezin zapis u direktoriju se zapravo ne briše nego mu se dodjeljuje posebna oznaka, a njezinim zapisima o clusterima u FAT tablici se dodjeljuju nule. Ta oznaka u direktoriju znači da se on može iskoristiti za neku drugu datoteku, ali omogućuje i vraćanje izbrisane datoteke. To naravno može biti samo moguće ako nije zapisana neka nova datoteka i ako clusteri nisu ponovno alocirani. [14]

4.2. NTFS

Kako su se računala i operacijski sustavi unaprjeđivali, tako FAT datotečni sustav više nije mogao ispuniti uvjete koje su oni zahtijevali. Ono što se tražilo od novog sustava je mogućnost oporavka, sigurnost datoteka, tolerancija grešaka te zalihost podataka. Tako je 1993. godine nastala prva verzija NTFS (*New Technology File System*) datotečnog sustava te je, počevši od Windows NT 3.1., NTFS zadani datotečni sustav Windows operacijskih sustava. [19]



Slika 4.7. *Primjer lokalnog diska s NTFS datotečnim sustavom (Windows 10)*

Kao i kod FAT32 datotečnog sustava, disk se može podijeliti na particije, a na svakoj particiji se nalaze clusteri u koje se pohranjuju podatci. Clusteri se sastoje od određenog broja sektora koji uvijek mora biti potencija broja 2, a veličina clustera ovisi o tome kolika je particija. S obzirom da je FAT32 bio razvijen u vrijeme malih particija, on nije bio optimalan za današnja računala zbog čega je i morao biti razvijen novi datotečni sustav. NTFS je prilagođen potrebama modernih računala pa podržava particije veličine i do 256TB, a veličina clustera je bitno prilagođena u odnosu na FAT32. Primjerice, veličina clustera za particiju veličine oko 300GB

kod FAT32 bila je 32KB, dok je kod NTFS-a samo 4KB. Sve veličine clustera mogu se vidjeti u tablici 4.3. [16]

Veličina particije	> MS Windows 2000
7 MB – 512 MB	4 KB
512 MB – 1 GB	4 KB
1 GB – 2 GB	4 KB
2 GB – 2 TB	4 KB
2 TB – 16 TB	4 KB
16 TB – 32 TB	8 KB
32 TB – 64 TB	16 KB
64 TB – 128 TB	32 KB
128 TB – 256 TB	64 KB
> 256 TB	Nije podržano

Tablica 4.3. *Zadane veličine clustera u NTFS datotečnom sustavu*

NTFS datotečni sustav sastoji se od tri dijela:

- *Partition Boot Sector*
- *Master File Table*
- Datotečno područje (engl. *File Area*)

NTFS datotečni sustav		
Partition Boot Sector	***Master File Table***	**Datotečno područje**

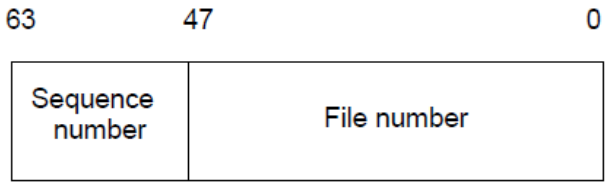
Prvi dio NTFS datotečnog sustava je *Partition Boot Sector* koji se nalazi na sektoru 0 te sadrži osnovne informacije potrebne za rad datotečnog sustava te je za njega osigurano 16 sektora. Informacije su slične kao i kod FAT32, primjerice broj bajtova po sektoru, broj sektora po clusteru, rezervirani sektori i slično. Također sadrži i lokaciju datoteke *Master File Table*, koji je ujedno i glavni dio NTFS-a.

Sve informacije o podatcima smještenima na disk nalaze se u datoteci *Master File Table* (MFT). Spremanjem svih podataka u tu datoteku, NTFS-u omogućuje jednostavan pronalazak datoteka, a sigurnosni deskriptor može zaštititi svaku datoteku. Sigurnosni deskriptor je ključan u sprječavanju neovlaštenog pristupa jer sadrži informaciju o vlasniku datoteke kao i dopuštenjima

koja je vlasnik dao drugim korisnicima. MFT zapravo čine nizovi zapisa, a u pojedinom zapisu su sadržani podatci o jednoj datoteci na disku. Također sadrži i zapis za samu sebe kako bi se mogla oporaviti u slučaju oštećenja. MFT ima zapise i za meta podatke NTFS datotečnog sustava. Nazivi meta podataka započinju sa znakom $ kako bi se lakše razlikovali od drugih sustavskih i korisničkih podataka. Neki od meta podataka su:

- $MFT – sama MFT tablica koja sadrži zapis za svaku datoteku ili direktorij na disku
- $MFTMirr – kopija najbitnijih dijelova MFT-a ukoliko dođe do njegovog gubitka
- $LogFile – ovdje se spremaju sve operacije koje su izvršene na disku, a koje mogu poslužiti u slučaju da dođe do neke greške, ovo se naziva *journaling* i veliki je napredak u odnosu na FAT32
- $Bitmap – pokazuje na slobodne clusteri
- $BadClus – pokazuje na oštećene clustere
- $Volume – sadrži podatke o particiji
- $AttrDef – *Attribute Definition Table*, definira tipove atributa koji su podržani te govori mogu li se oporaviti ukoliko to bude potrebno

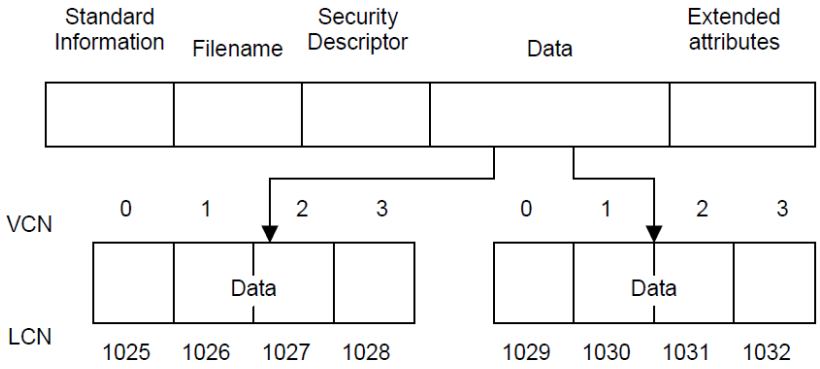
Kako bi se NTFS mogao pozivati na te zapise, svaki zapis ima jedinstvenu 64-bitnu ID vrijednost koja se naziva referenca datoteke (engl. *File Reference*). Ona se sastoji od broja datoteke (engl. *File Number*), koji određuje poziciju u MFT tablici, i rednog broja (engl. *Sequence Number*), koji se poveća svaki put kad se zapis ponovno alocira (slika 4.8.).



Slika 4.8. *File Reference*

Svaki red u MFT nizu predstavlja zapis neke datoteke, a zapis je zapravo skup atributa i njihovih vrijednosti. Neki od tih atributa su naziv datoteke, sigurnosni deskriptor te sami podatci neke datoteke. Takav način zapisa omogućuje brzi pristup datotekama, pogotovo ako se radi o manjim datotekama. Primjerice, kod FAT datotečnog sustava, koji koristi tablicu za pristup podatcima, potrebno je prvo proći FAT tablicu kako bi se pronašle sve pozicije datoteke i kako bi se ona u konačnici mogla dohvatiti. Kod NTFS-a je ovaj proces puno jednostavniji jer je potrebno samo naći datoteku u MTF tablici i tamo se već nalaze podatci o njoj.

Dakle, podatci o nekoj datoteci su također atribut i kod manjih datoteka se u cjelosti spremaju u zapis MTF-a te se tada taj atribut naziva *resident*. Kod većih datoteka, kada podatci o njoj ne mogu stati u zapis MFT-a, radi se o *nonresident* atributu. U tom slučaju se u zapis sprema pokazivač koji pokazuje na dio diska (engl. *data run*, *data extent*) na kojemu se nalazi vrijednost određenog atributa. Primjer takvog slučaja je vidljiv na slici 4.9.



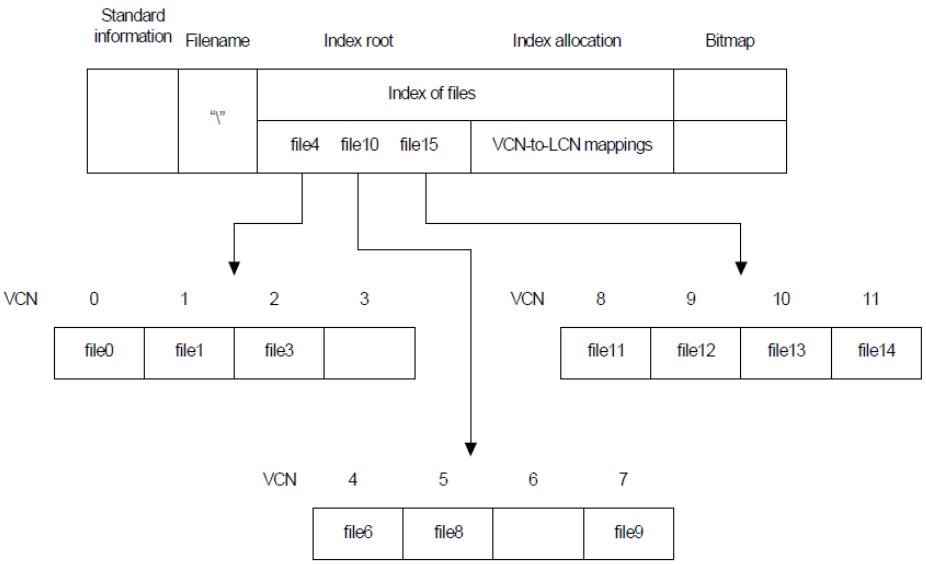
Slika 4.9. *MFT zapis s nonresident atributom*

Dakle, podatci datoteke spremljeni su na pojedine dijelove diska, a kako bi se znala pozicija tih dijelova, tj. kako bi im se moglo pristupiti koriste se logički brojevi clustera (engl. *logical cluster numbers* – LCN). Logički brojevi su zapravo samo brojčana oznaka clustera koju ima svaki cluster na particiji. Osim LCN-a, potreban je i virtualni broj clustera (engl. *virtual cluster number* – VCN), a on govori koji clusteri pripadaju određenoj datoteci s obzirom da datoteka može biti smještena na nekoliko clustera. Dakle, kada se disk podijeli na clustere, svaki cluster se redom numerira, počevši od 0 i to predstavlja LCN, a svaka datoteka je smještena na nekoliko od tih clustera i, kako bi se znalo koji cluster je početni te koji svi clusteri pripadaju nekoj datoteci, svaki cluster datoteke se numerira, također počevši od 0 i to predstavlja VCN.

Osim zapisa datoteka, u MFT-u su spremljeni i zapisi o direktorijima. Direktorij čine samo imena datoteka s referencama na njihove zapise, a njegova svrha je organizacija koja omogućuje brži i jednostavniji pristup. Sve počinje od *Root* direktorija čija se pozicija nalazi također u MFT tablici. Na slici 4.10. se može vidjeti primjer *Root* direktorija. Svaki zapis direktorija sadrži *index root* atribut u kojem se nalazi sortirana lista datoteka u direktoriju. Kod većih direktorija, lista datoteka se nalazi u *index buffer*-ima, koji sadrže i organiziraju nazive datoteka.

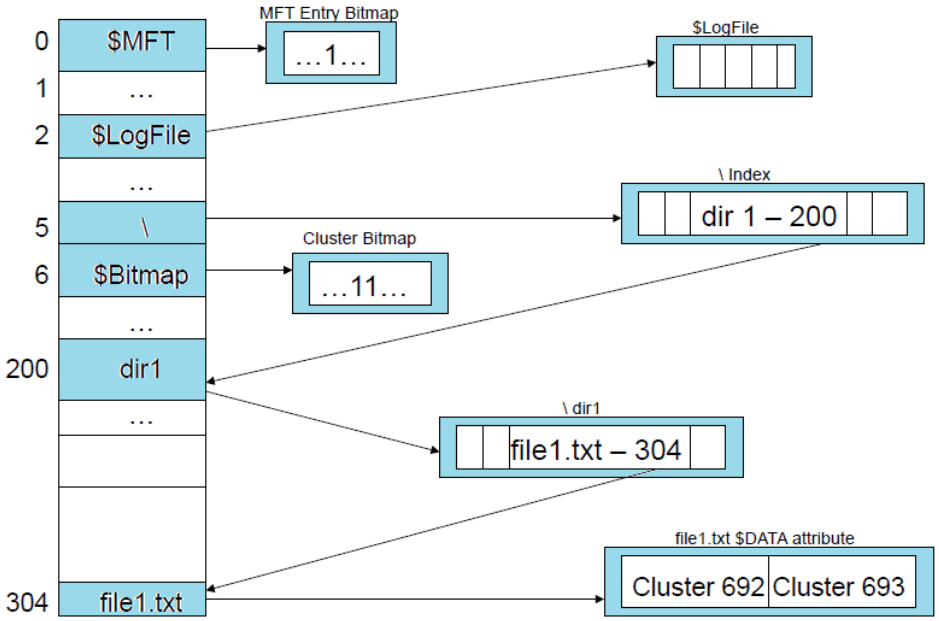
Index root atribut sadrži poddirektorije i pokazuje na njihove *index buffere*. *Index allocation* atribut daje informacije o LCN i VCN vrijednostima kako bi se mogla utvrditi lokacija *index buffera*. Bitmap atribut govori koji VCN-ovi u *index bufferima* su slobodni a koji nisu. [20]

Za pohranu, pretragu i brisanje zapisa o direktorijima NTFS koristi strukturu koja se naziva *B+ tree*. Struktura binarnog stabla korisna je kod organiziranja velike količine podataka jer ih pri pohrani sortira te smanjuje broj potrebnih pristupa disku da bi se pronašao određeni podatak. Sortiranje se odvija na način da su lijevo uvijek manje, a desno uvijek veće vrijednosti od postojećeg čvora. [21]



Slika 4.10. *Root direktorij*

Pri pohrani nove datoteke alocira se mjesto za novi zapis u MFT tablici te se u taj zapis unose osnovne informacije, tj. atributi. Zatim se provjerava meta podatak $Bitmap kako bi se pronašli slobodni clusteri te se uglavnom popunjavaju ranije spomenutim *best-fit* algoritmom. Bitmapima se mijenja vrijednost iz 0 u 1. Podatci se smještaju u clustere te se u MFT zapis unose vrijednosti o pozicijama clustera. Pomoću *Root* direktorija pronalazi se određeni direktorij te se smješta zapis o datoteci. Ovaj postupak se može vidjeti na slici 4.11. Kod brisanja datoteke, pronalazi se određeni direktorij te se briše njezin zapis. Također se oslobađa zapis o toj datoteci i u MFT-u, a bitmapima se vraća vrijednost u 0. Svi koraci se spremaju u $LogFile. [14]



Slika 4.11. *Postupak pohrane nove datoteke*

5. USPOREDBA FAT32 I NTFS DATOTEČNIH SUSTAVA

Kao što je već rečeno, NTFS je nastao zbog toga što FAT32 nije zadovoljavao uvjete koje su postavljala naprednija računala te njihova šira primjena. Osim što se ova dva sustava razlikuju u samom principu rada, odnosno načinu pohrane i pretrage podataka, što je objašnjeno u prethodnim poglavljima, razlikuju se i u maksimalnoj dopuštenoj veličini particije i datoteke te maksimalnom dopuštenom broju datoteka i slično. Jednostavna usporedba prikazana je u tablici 5.1. Iz tablice je vidljivo da je NTFS uveo neke nove značajne karakteristike, kao što su zapisi i sigurnost, a koje su pobliže objašnjene u nastavku. Dodatne karakteristike ga čine složenijim sustavom u odnosu na FAT32 te zbog toga konverzija iz NTFS sustava u FAT32 nije dopuštena, dok je obrnuto moguće. [22][23]

	FAT32	NTFS
Alokacija datoteke	Povezane liste	Bitmap
Sadržaj direktorija	Tablica	Binarno stablo (engl. *B-tree variant*)
Oštećeni blokovi	Označavanje clustera	$BadClus (MFT zapis)
Max. veličina particije	2 TiB (sa sektorima od 512 bajtova) 8 TiB (sa sektorima od 2 KiB i clusterima od 32 KiB) 16 TiB (sa sektorima od 4 KiB i clusterima od 64 KiB)	256 TiB (s clusterima od 64 KB; Windows 10 verzija 1703, Windows Server 2016 ili ranija implementacija) 8 PB (s clusterima od 2 MB; Windows 10 verzija 1709, Windows Server 2019 ili kasnije)
Max. veličina datoteke	2,147,483,647 B (2 GiB – 1) (bez LFS – *Large-file support*) 4,294,967,295 B (4 GiB – 1) (sa LFS-om) 274,877,906,943 B (256 GiB – 1) (samo sa FAT32+)	16 TB (s clusterima od 4 KB; Windows 7, Windows Server 2008 R2 ili ranije) 256 TB (s clusterima od 64 KB; Windows 8, Windows Server 2012 ili kasnije) 8 PB (s clusterima od 2 MiB; Windows 10 verzija 1709, Windows Server 2019 ili kasnije)
Max. broj datoteka	268,173,300 za clustere od 32 KiB	4,294,967,295 (2^{32}-1)
Max. duljina naziva datoteke	*8.3 filename* sa OEM znakovima, 255 UCS-2 znakova korištenjem LFN-a (*Long filename)*	255 UTF-16 *code units*

Max. dubina direktorija	32 razine ili 66 znakova (sa CDS – *Current Directory Structure*), 60 razina ili više (bez CDS-a)	Nije ograničeno
Kompresija	Nema	Ima (LZNT1 algoritam)
Enkripcija	Nema	Ima (*Encrypting File System*)
Konverzija	Dopušteno	Nije dopušteno
Tolerancija grešaka	Nema	Automatsko rješavanje grešaka
Zapisi	Nema	Ima (*Journaling File System*)
Sigurnost	Nema	Ima (ACLs)
Kompatibilnost s ostalim operacijskim sustavima	Windows, Linux, Mac	Windows, Linux – čitanje i pisanje moguće samo s dodatnim programom (ntfs-3g), Mac – samo čitanje, a pisanje moguće s dodatnim programom

Tablica 5.1. *Usporedba FAT32 i NTFS datotečnih sustava*

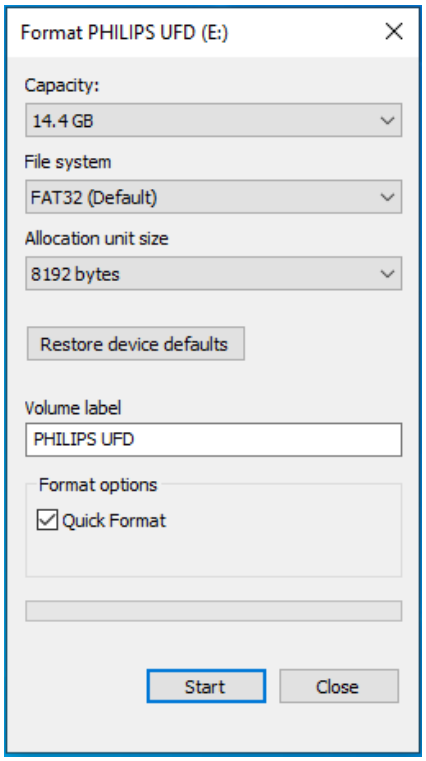
NTFS ima mogućnost oporavka na način da nijedna operacija ne može ostati nedovršena, odnosno sve se operacije moraju izvršiti do kraja. Kada se operacija uspješno izvrši, pravi se zapis koji pamti to stanje. Za zapise (engl. *journaling*) NTFS koristi već spomenuti $LogFile. U slučaju da dođe do pada sustava ili do iznenadnog nestanka struje usred izvršavanja neke operacije, sve dotad napravljeno se poništava i vraća na stanje prije početka izvođenja operacije što sprječava potencijalno oštećenje podataka.

Kod NTFS-a svaka datoteka ima svoj sigurnosni deskriptor, koji je spomenut u prošlom poglavlju, a sastoji se od dvije kontrolne liste pristupa (engl. *Access control lists* - ACLs). Prvi ACL je *Discretionary access control list* (DACL) koji točno definira radnje (npr. čitanje, pisanje, brisanje) koje su dopuštene ili zabranjene pojedinom korisniku. Drugi ACL naziva se *System access control list* (SACL) koji bilježi kada je netko htio obrisati ili kopirati neku datoteku te je li u tome uspio, a takvi zapisi su korisni primjerice u tvrtkama koje imaju povjerljive podatke. [24]

Unatoč svim poboljšanjima NTFS sustava, FAT32 u pravilu je i dalje bolji izbor datotečnog sustava za prijenosne medije, primjerice za USB memoriju. Njegova najveća prednost je to što je zbog svoje jednostavne strukture kompatibilan sa gotovo svim operacijskim sustavima. Nove karakteristike koje pruža NTFS mogu zapravo biti nedostatak u slučaju prijenosnih medija. Na primjer, s obzirom da zapisi zahtijevaju dodatno pisanje na medij, oni ga mogu samo dodatno istrošiti jer takva vrsta medija ima ograničeni broj pisanja. Također, mehanizam sigurnosti može zabraniti kopiranje ili brisanje pojedine datoteke kada joj pokušamo pristupiti s drugog računala. Mala prednost u performansama NTFS sustava u odnosu na FAT32 nije toliko važna u ovom slučaju s obzirom da prijenosne medije uglavnom ne koristimo svakodnevno, niti za prijenos velikih i važnih datoteka. [25]

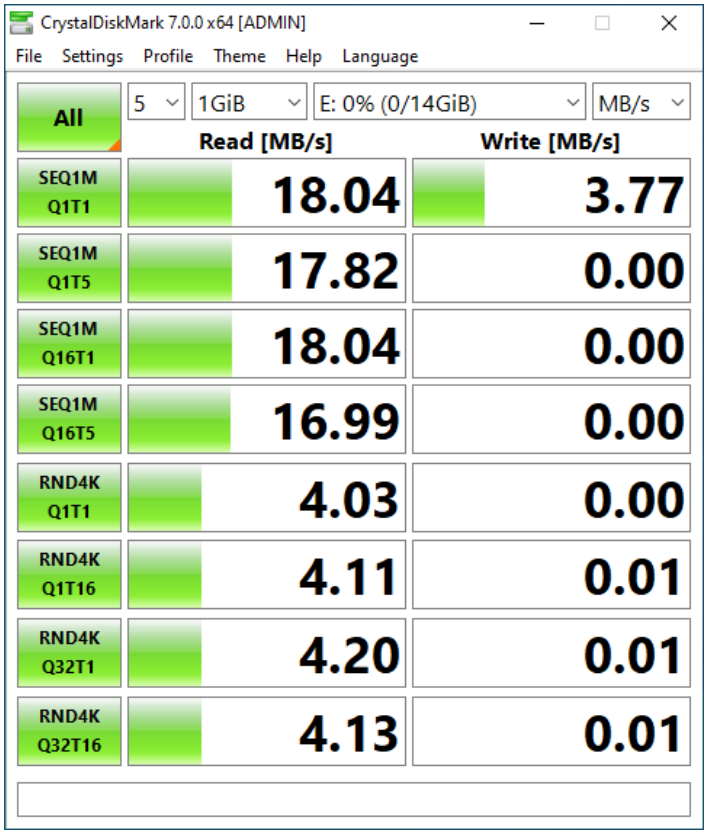
5.1. Testiranje performansi na USB sticku

U svrhu usporedbe u ovom poglavlju provedeno je testiranje performansi FAT32 i NTFS datotečnih sustava na USB sticku pomoću CrystalDiskMark alata te na način da se izmjeri koliko vremena je potrebno da se na stick prenese jedna datoteka od gotovo 4GB te 100 datoteka po 40MB koje se nalaze u jednom direktoriju. Također je izmjereno i koliko vremena je potrebno za brisanje tih datoteka. Prvo je napravljeno testiranje FAT32 sustava te je u skladu s tim USB stick formatiran s FAT32 datotečnim sustavom, što je vidljivo na slici 5.1.



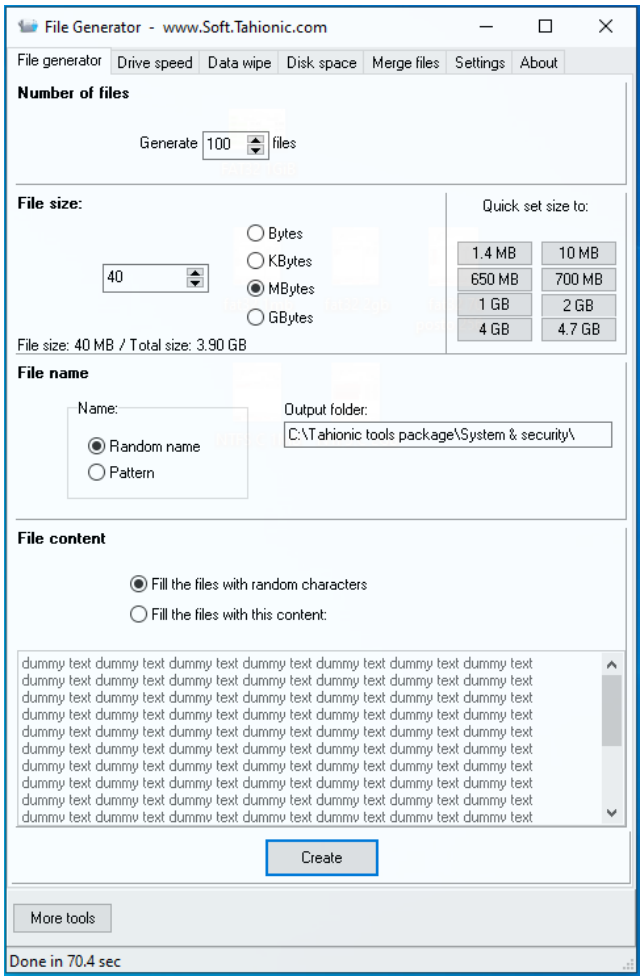
Slika 5.1. *Formatiranje USB sticka s FAT32 datotečnim sustavom*

Nakon toga izvršeno je testiranje performansi USB sticka pomoću CrystalDiskMark alata. Alat je besplatan te omogućuje mjerenje brzina operacija čitanja (engl. *read*) i pisanja (engl. *write*). Testiranje je moguće sekvencijalnim (engl. *sequential* - SEQ) i nasumičnim (engl. *random* - RND) pristupom, te je moguće postaviti nekoliko parametara, kao što su redovi (engl. *queues* - Q) i niti (engl. *threads* - T). U ovom slučaju izabrane su različite kombinacije za sekvencijalni i nasumični pristup, a testna datoteka je veličine 1GB. Rezultati testiranja vidljivi su na slici 5.2. Može se primjetiti da je sekvencijalni pristup znatno brži od nasumičnog te da kod sekvencijalnog pristupa što je više niti, to su operacije sporije.



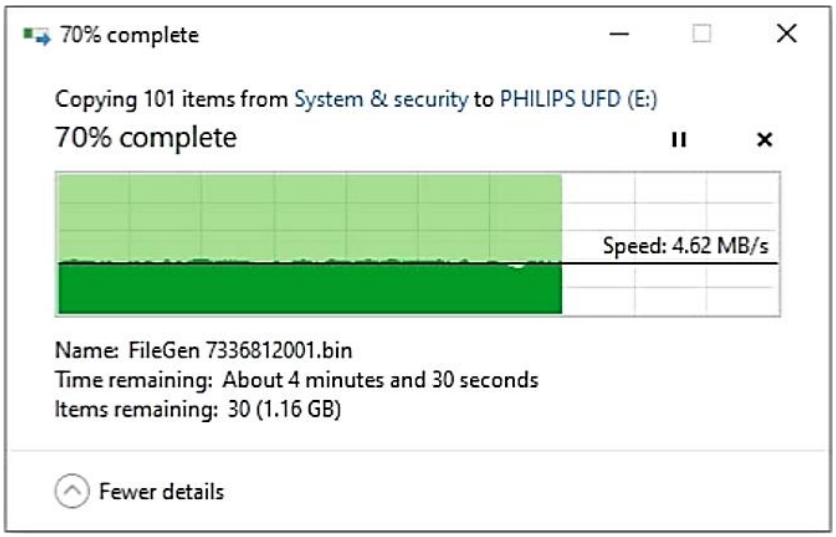
Slika 5.2. *Brzina write/read operacija na USB sticku s FAT32 sustavom*

Zatim je proveden test mjerenja vremena potrebnog za prijenos datoteka na USB stick, jedne datoteke od 4GB i 100 datoteka od 40MB, a koje se nalaze u jednom direktoriju i zajedno iznose 4GB. Datoteke su kreirane pomoću alata File Generator koji omogućuje kreiranje određenog broja datoteka odabrane veličine što je vidljivo na slici 5.3.

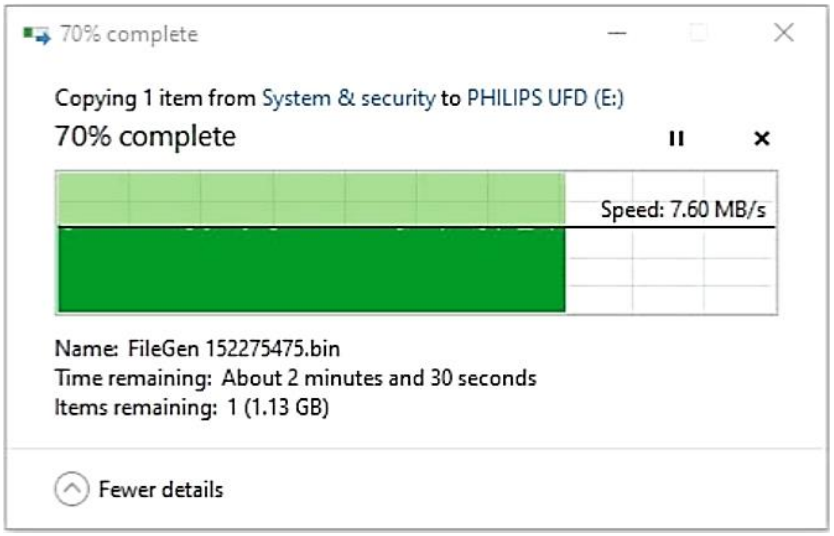


Slika 5.3. *Kreiranje datoteka pomoću alata File Generator*

Rezultat prijenosa 100 datoteka od 40MB koje se nalaze u jednom direktoriju je da je u većini slučajeva postignuta brzina oko 4.80 MB/s, a iznimno je brzina išla i do 7 MB/s. Vrijeme potrebno za prijenos variralo je između 9 i 14 minuta. U konkretnom slučaju, prikazanom na slici 5.4., za prijenos datoteka potrebno je bilo 14 minuta što znači da je prosječna brzina prijenosa bila 4.35 MB/s. Rezultat prijenosa jedne datoteke od 4GB je da je postignuta brzina uglavnom bila oko 7.70 MB/s, dok je vrijeme potrebno za prijenos između 8 min. i 40 s te 9 min. i 10 s. U slučaju prikazanom na slici 5.5. za prijenos datoteke potrebno je bilo 8 minuta i 50 sekundi što znači da je prosječna brzina za prijenos bila 7.30 MB/s. Dakle, vidljivo je da je manje vremena potrebno da se prenese jedna veća datoteka od 4GB nego više manjih koje zajedno čine 4GB zbog toga što se u pravilu postigne veća brzina prijenosa u slučaju jedne veće nego više manjih datoteka.

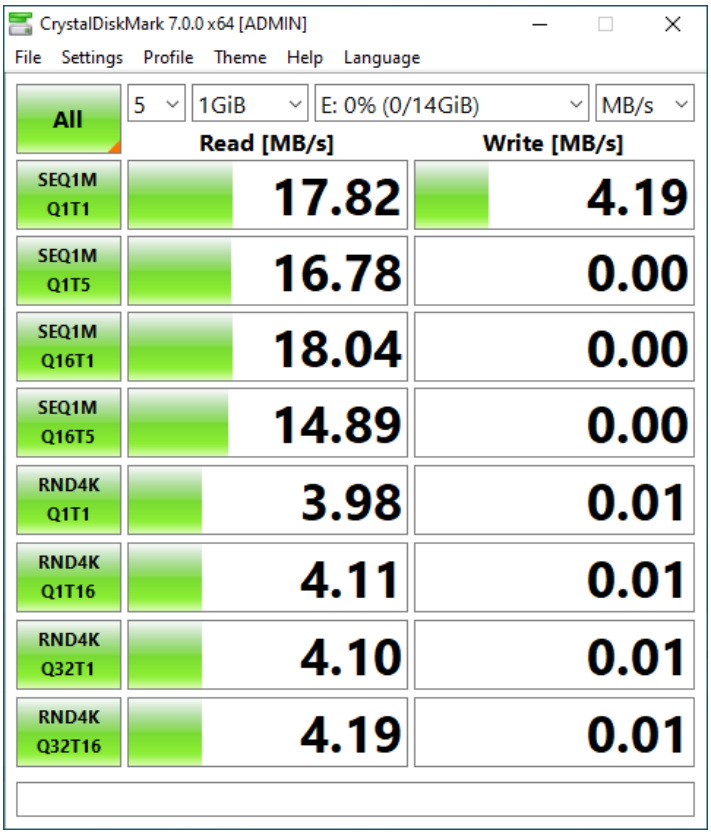


Slika 5.4. *Prijenos 100 datoteka od 40MB koje se nalaze u jednom direktoriju*



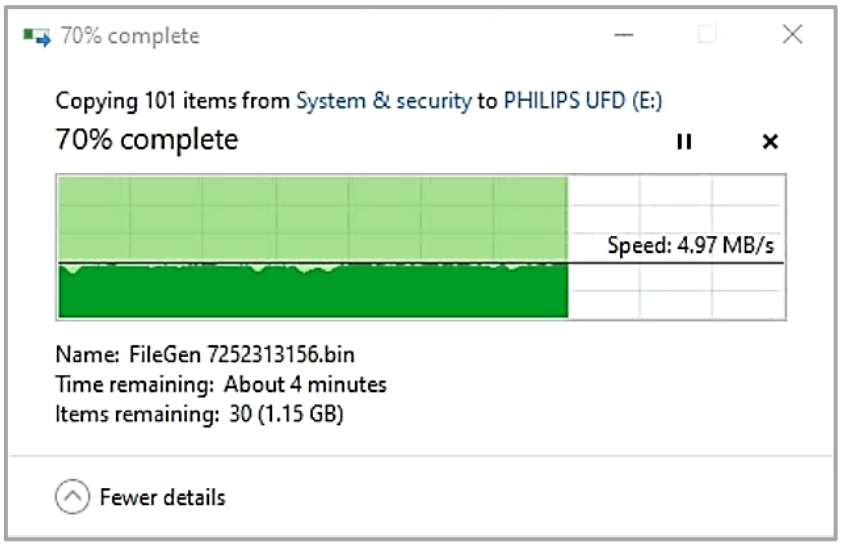
Slika 5.5. *Prijenos 1 datoteke od 4GB*

Nakon provedenog testiranja FAT32 datotečnog sustava, svi ovi koraci ponovljeni su na USB sticku koji je formatiran s NTFS datotečnim sustavom. Dakle, prvo je izvršeno testiranje performansi USB sticka pomoću CrystalDiskMark alata čije rezultate možemo vidjeti na slici 5.6. Može se primjetiti da su dobiveni rezultati dosta slični kao i kod FAT32 sustava.

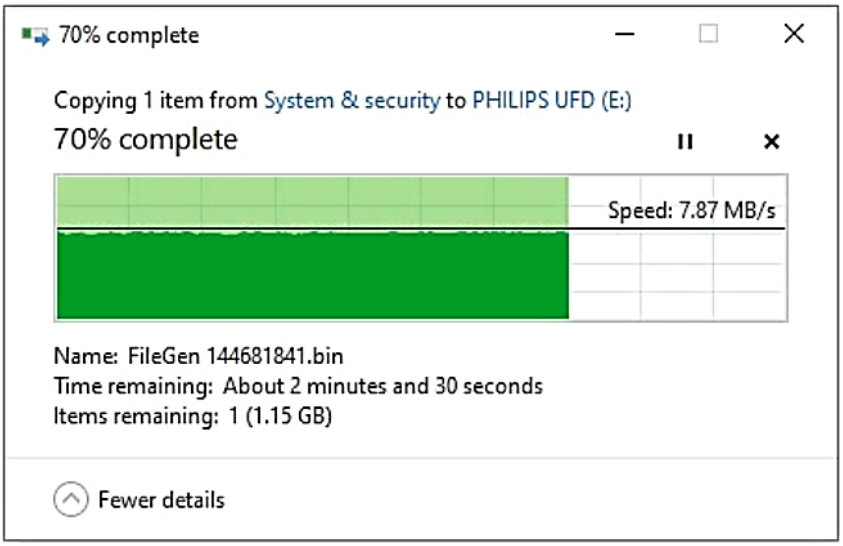


Slika 5.6. *Brzina write/read operacija na USB sticku s NTFS sustavom*

Zatim je, kao i u prethodnom slučaju, proveden test mjerenja vremena potrebnog za prijenos datoteka na USB stick, jedne veće i više manjih datoteka. Rezultat prijenosa 100 datoteka od 40MB koje se nalaze u jednom direktoriju je sličan kao i kod FAT32 sustava, što znači da je u većini slučajeva postignuta brzina oko 4.70 MB/s, a iznimno je brzina išla i preko 7 MB/s. Vrijeme potrebno za prijenos također je variralo između 9 i 14 minuta. U slučaju prikazanom na slici 5.7. za prijenos datoteka potrebno je bilo 14 minuta što znači da je prosječna brzina za prijenos bila 4.35 MB/s. Rezultat prijenosa jedne datoteke od 4GB je da je postignuta brzina uglavnom bila oko 7.80 MB/s, dok je vrijeme potrebno za prijenos bilo u pravilu 8 min. i 30 s, kao što je to i u slučaju prikazanom na slici 5.8. To znači da je prosječna brzina za prijenos bila 7.47 MB/s. Dakle, i kod NTFS-a je također vidljivo da se brže prenese jedna veća datoteka od 4GB nego više manjih koje zajedno čine 4GB.



Slika 5.7. *Prijenos 100 datoteka od 40MB koje se nalaze u jednom direktoriju*



Slika 5.8. *Prijenos 1 datoteke od 4GB*

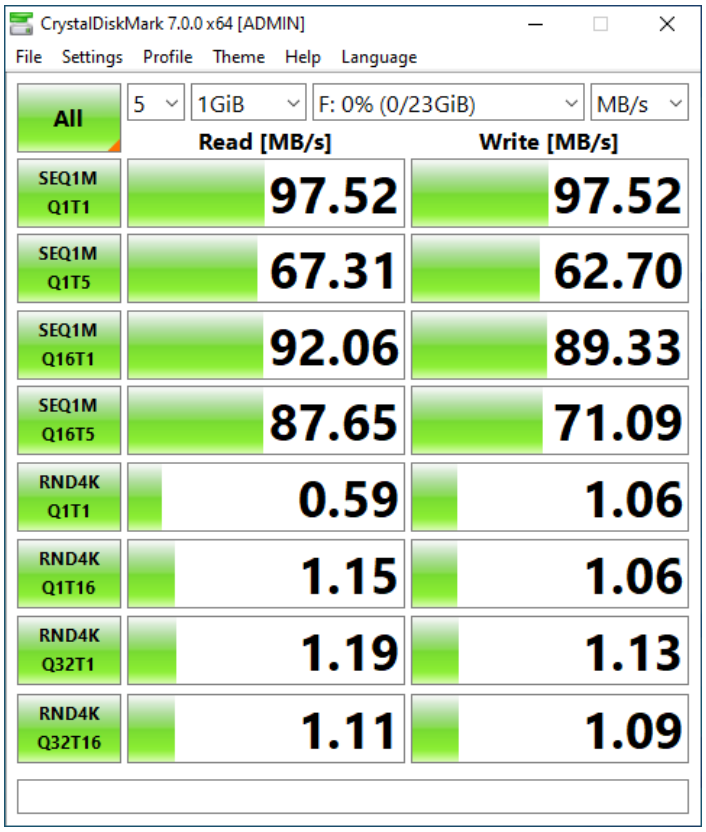
U svrhu testiranja brzine pojedinog datotečnog sustava, mjereno je i vrijeme potrebno za brisanje datoteka. U tom slučaju može se primjetiti značajna razlika, odnosno za brisanje jedne veće datoteke kod FAT32 sustava potrebno je par sekundi, dok se brisanje iste datoteke kod NTFS sustava događa u istom trenutku. Za brisanje više manjih datoteka kod FAT32 sustava potrebno je bilo oko 8 sekundi, a kod NTFS sustava samo 3 sekunde.

Može se zaključiti da zapravo nema značajne razlike u brzini samog izvođenja operacije pisanja između FAT32 i NTFS datotečnih sustava na USB sticku. Međutim, da bi se uopće započeo proces pisanja, kod FAT32 sustava potrebno je bilo oko 1-3 sekunde, ako se radilo o više manjih datoteka, a oko 5 sekundi, ako se radilo o jednoj većoj datoteci, dok je kod NTFS sustava proces pisanja započinjao odmah. Isto tako, kod operacije brisanja može se primjetiti bolja performansa NTFS sustava u odnosu na FAT32. Sve to je rezultat MFT zapisa te strukture binarnog stabla koje koristi NTFS, a koji su brži u odnosu na tablicu koju koristi FAT32.

5.2. Testiranje performansi na tvrdom disku

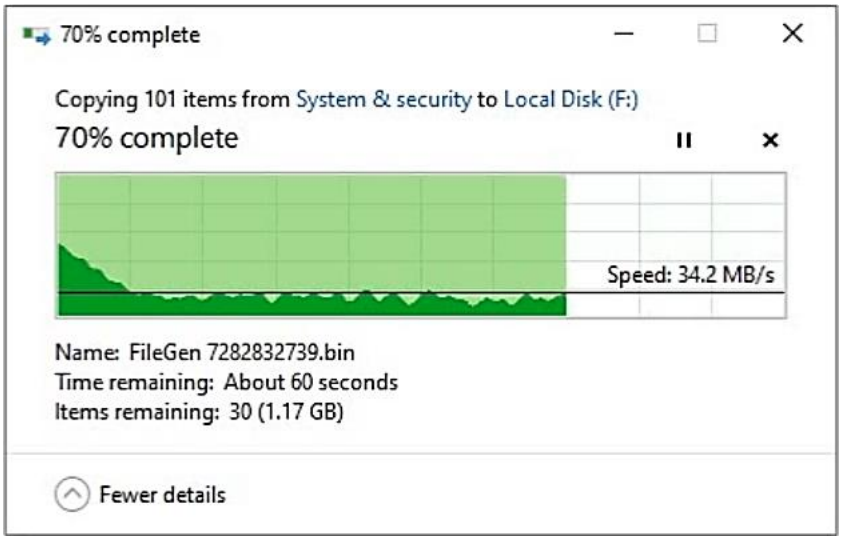
U ovom poglavlju ponovno je provedeno testiranje performansi FAT32 i NTFS datotečnih sustava, ali na tvrdom disku. Testiranje je također provedeno pomoću CrystalDiskMark alata te na način da se izmjeri koliko vremena je potrebno da se na disk prenese jedna veća datoteka i 100 manjih datoteka. Također je izmjereno i koliko vremena je potrebno za brisanje tih datoteka.

Prvo je tvrdi disk formatiran s FAT32 datotečnim sustavom te je izvršeno testiranje performansi pomoću CrystalDiskMark alata čije rezultate možemo vidjeti na slici 5.9.



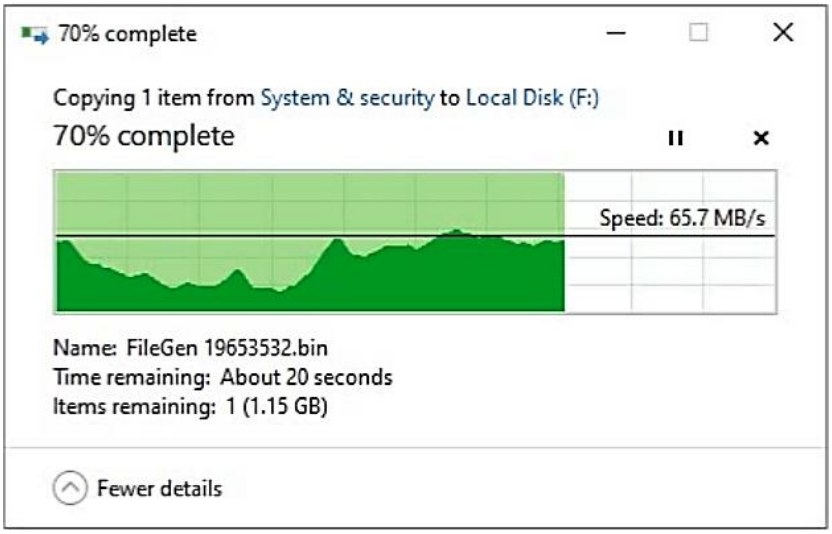
Slika 5.9. *Brzina write/read operacija na tvrdom disku s FAT32 sustavom*

Zatim je proveden test mjerenja vremena potrebnog za prijenos datoteka na tvrdi disk, jedne veće i više manjih datoteka. Rezultat prijenosa 100 datoteka od 40MB koje se nalaze u jednom direktoriju je da je u većini slučajeva postignuta brzina bila između 20 MB/s i 35 MB/s. Vrijeme potrebno za prijenos bilo je oko 2 min i 15 s, kao što je to u slučaju prikazanom na slici 5.10. To znači da je prosječna brzina za prijenos bila 29 MB/s.



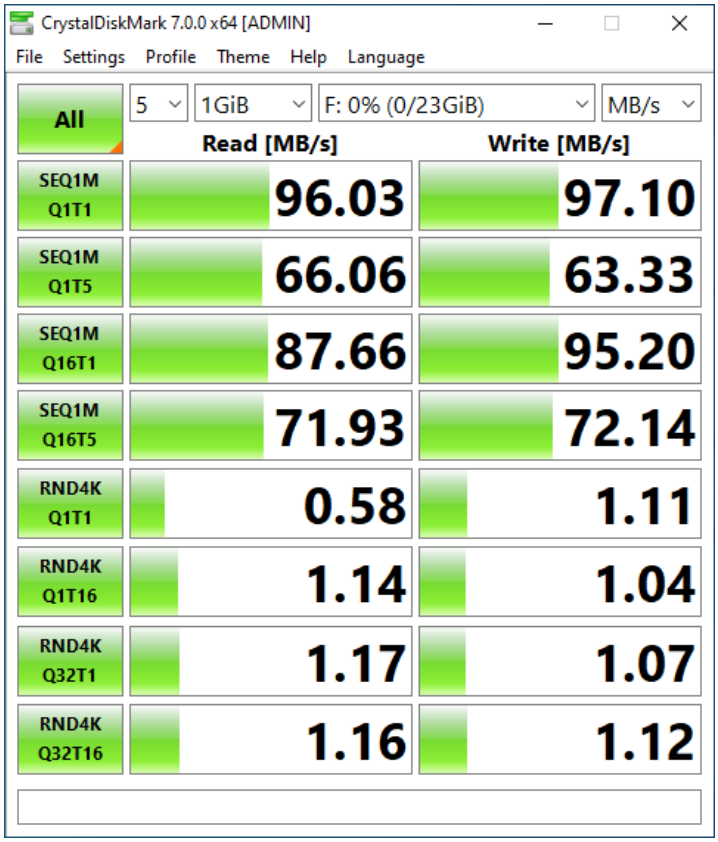
Slika 5.10. *Prijenos 100 datoteka od 40MB koje se nalaze u jednom direktoriju*

Rezultat prijenosa jedne datoteke od 4GB je da je postignuta brzina bila između 20 MB/s i 70 MB/s, dok je vrijeme potrebno za prijenos bilo oko 1 min. 30 s, kao što je prikazano na slici 5.11. To znači da je prosječna brzina za prijenos bila 48 MB/s. Slično kao i kod USB sticka, u većini slučajeva brže se prenese jedna veća datoteka od 4GB nego više manjih koje zajedno čine 4GB, također zbog toga što se u pravilu postigne veća brzina prijenosa jedne veće datoteke nego više manjih datoteka.



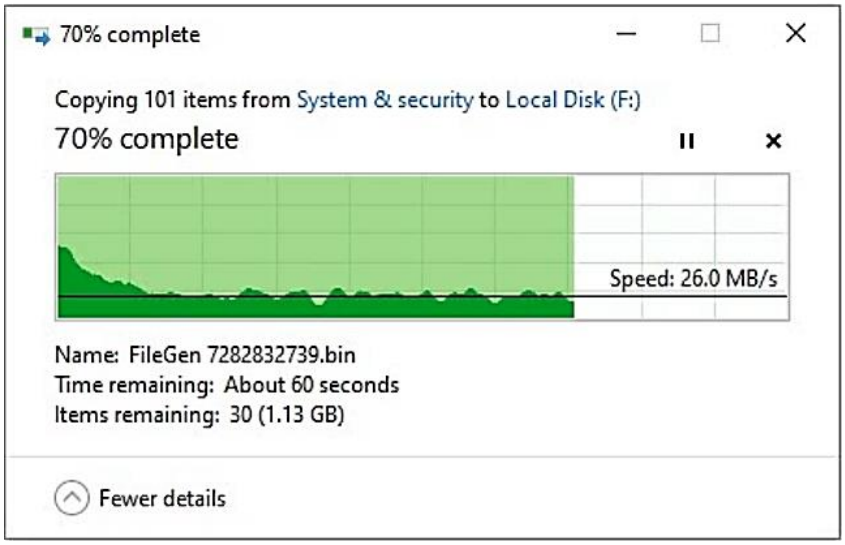
Slika 5.11. *Prijenos 1 datoteke od 4GB*

Nakon provedenog testiranja FAT32 datotečnog sustava, svi ovi koraci ponovljeni su na tvrdom disku koji je formatiran s NTFS datotečnim sustavom. Dakle, prvo je izvršeno testiranje performansi tvrdog diska pomoću CrystalDiskMark alata čije rezultate možemo vidjeti na slici 5.12. Može se primjetiti da se dobiveni rezultati ne razlikuju značajno od FAT32 sustava.



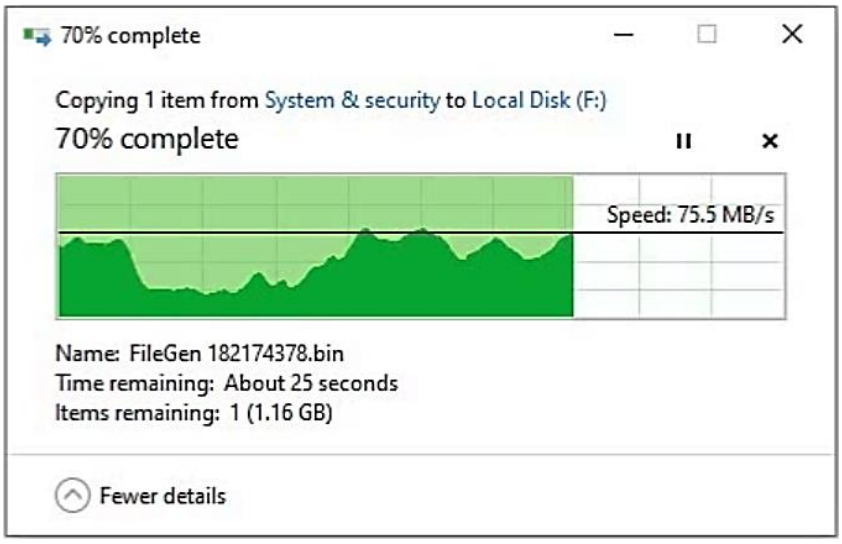
Slika 5.12. *Brzina write/read operacija na tvrdom disku s NTFS sustavom*

Kao i u prethodnom slučaju, proveden je test mjerenja vremena potrebnog za prijenos datoteka na tvrdi disk. Za prijenos 100 manjih datoteka koje se nalaze u jednom direktoriju potrebno je bilo između 1 min. 45 s i 2 min. 25 s, a postignuta brzina varirala je od 25 MB/s do 35 MB/s. U slučaju prikazanom na slici 5.13. za prijenos su potrebne bile 2 min. 15 s, što znači da je prosječna brzina bila 29.5 MB/s.



Slika 5.13. *Prijenos 100 datoteka od 40MB koje se nalaze u jednom direktoriju*

Kod prijenosa jedne veće datoteke postignuta je brzina između 25 MB/s i 75 MB/s. Prijenos je u pravilu trajao oko 1 min 25 s, kao što je to i u slučaju prikazanom na slici 5.14, što znači da je prosječna brzina bila 49 MB/s. Može se primjetiti da su rezultati slični kao kod FAT32 sustava.



Slika 5.14. *Prijenos 1 datoteke od 4GB*

Na kraju je izmjereno vrijeme potrebno za brisanje datoteka na tvrdom disku. Između dva datotečna sustava nije bilo razlike, odnosno potrebne su bile do oko 2 sekunde za brisanje i jedne veće i više manjih datoteka. Također, kod oba sustava sve operacije su započinjale odmah.

Testiranjem se zaključuje da nema znatne razlike u brzini započinjanja i izvođenja operacija između FAT32 i NTFS datotečnih sustava na tvrdom disku, uglavnom zbog toga što je ona danas, zbog brzih CPU-a pri operacijama čitanja i pisanja, ograničena brzinom diska. Također, budući da danas nema nekakvog penala na performanse, bolje je koristiti NTFS, zbog mogućnosti i unaprijeđenja u odnosu na FAT32, što se tiče sigurnosti, pouzdanosti i sl.

U nastavku je prikazana tablica 5.2. koja daje pregled svih dobivenih rezultata testiranja.

	FAT32	**NTFS**
USB stick		
Prijenos više manjih datoteka	Između 9 i 14 min.	Između 9 i 14 min.
Postignuta brzina	4.80 MB/s, iznimno do 7 MB/s	4.70 MB/s, iznimno do 7 MB/s
Započinjanje operacije	1-3 sekunde	Trenutno
Brisanje	8 sekundi	3 sekunde
Prijenos jedne veće datoteke	Između 8 min. 40 s i 9 min. 10 s	Oko 8 min. i 30 s
Postignuta brzina	7.70 MB/s	7.80 MB/s
Započinjanje operacije	5 sekundi	Trenutno
Brisanje	Oko 2 sekunde	Trenutno
Tvrdi disk		
Prijenos više manjih datoteka	Oko 2 min. 15 s	Između 1 min. 45 s i 2 min. 25 s
Postignuta brzina	Između 20 MB/s i 35 MB/s	Između 25 MB/s i 35 MB/s
Brisanje	Oko 2 sekunde	Oko 2 sekunde
Prijenos jedne veće datoteke	Oko 1 min. 30 s	Oko 1 min. 25 s
Postignuta brzina	Između 20 MB/s i 70 MB/s	Između 20 MB/s i 75 MB/s
Brisanje	Oko 2 sekunde	Oko 2 sekunde
Započinjanje operacija	Trenutno	Trenutno

Tablica 5.2. *Pregled prosječnih rezultata testiranja*

6. ZAKLJUČAK

Način rada današnjih računala ne bi bio moguć bez trajne pohrane podataka. Ona omogućuje spremanje bitnih podataka koji će i kasnije biti potrebni što znači da se u većini slučajeva radi o velikoj količini podataka. Sve to pruža trajna memorija, poput HDD-a ili SSD-a. Međutim, potreban je bio način kako te podatke smjestiti u cjeline, a te cjeline smjestiti u memoriju i kako na kraju pronaći određeni podatak. To je riješeno uvođenjem datoteka i direktorija, gdje datoteka predstavlja podatke organizirane u cjelinu, a direktorij služi za dodatnu organizaciju datoteka. Za pohranu datoteka i direktorija na disk i njihovo kasnije dohvaćanje brine se datotečni sustav.

Opisana su dva najpoznatija datotečna sustava u Windows računalima – FAT32 i NTFS. Princip rada FAT32 (*32-bit File Allocation Table*) sustava temelji se na tablici koja sadrži zapise jednakih veličina, a broj zapisa odgovara broju clustera na disku. Clusteri su cjeline koje se sastoje od određenog broja sektora, a neki od zapisa mogu biti: slobodan cluster, sljedeći cluster u lancu, kraj datoteke. Za pristup podatcima koristi se struktura povezane liste što znači da tablica sadrži zapis o svakom sljedećem clusteru datoteke na disku. Zbog toga se ne mora pretraživati cijeli disk nego samo ta tablica, a tek se onda pristupa tim traženim podatcima na disku. Struktura direktorija sadrži informacije o početnom clusteru svake datoteke.

NTFS (*New Technology File System*) temelji se na MFT (*Master File Table*) tablici u kojoj se nalaze sve informacije o podatcima smještenima na disk te je zbog toga pronalazak datoteka jednostavan. Svaki red u MFT-u predstavlja zapis neke datoteke, a koji je zapravo skup atributa, kao što su naziv datoteke, sigurnosni deskriptor te pokazivač na dijelove diska na kojima se nalazi datoteka. Za oznaku svakog clustera na disku koristi se logički broj clustera (LCN), a za oznake clustera koji pripadaju pojedinoj datoteci koristi se virtualni broj clustera (VCN). Za pretragu NTFS koristi strukturu binarnog stabla (*B+tree*) što je korisno kod organiziranja velike količine podataka jer je njihov pronalazak puno brži.

Zbog ograničenja te manjka sigurnosti i pouzdanosti koje ima FAT32, NTFS je s razlogom zadani datotečni sustav Windows operacijskih sustava. Međutim, upravo zbog svoje jednostavnosti, FAT32 je dobar izbor datotečnog sustava za prijenosne medije. Testiranjem performansi ova dva sustava na USB sticku može se zaključiti da u samom započinjanju operacija općenito te u izvođenju operacije brisanja prednost ima NTFS, dok kod izvođenja operacije pisanja nema značajne razlike. Testiranje performansi na tvrdom disku pokazalo je da su razlike u izvođenju operacija između FAT32 i NTFS sustava zanemarive.

LITERATURA

[1] A. Tanenbaum, Modern Operating Systems, Pearson Education, Amsterdam, 2009.

[2] https://www.ufsexplorer.com/articles/file-systems-basics.php (25.5.2019.)

[3] https://docs.microsoft.com/hr-hr/windows/desktop/FileIO/naming-a-file (12.6.2019.)

[4] https://www.digitalcitizen.life/what-file-s-metadata-and-how-edit-it (12.6.2019.)

[5] https://en.wikipedia.org/wiki/File_attribute (13.6.2019.)

[6] https://www.geeksforgeeks.org/operating-system-structures-of-directory/ (21.6.2019.)

[7] https://www.computerhope.com/issues/ch001708.htm (29.6.2019.)

[8] R. Arpaci-Dusseau, A. Arpaci-Dusseau, Operating Systems: Three Easy Pieces, Arpaci-Dusseau Books, Madison-Wisconsin, 2015.

[9] http://www.active-undelete.com/hdd_basic.htm (19.12.2019.)

[10] https://hr.wikipedia.org/wiki/Tvrdi_disk (19.12.2019.)

[11] https://en.wikipedia.org/wiki/File_Allocation_Table (6.10.2019.)

[12] http://www.cs.fsu.edu/~cop4610t/assignments/project3/spec/fatspec.pdf (17.10.2019.)

[13] https://en.wikipedia.org/wiki/Design_of_the_FAT_file_system (6.10.2019.)

[14] http://topdownbook.com/uploads/1/0/8/4/108471219/filesystemforensics.pdf (17.10.2019.)

[15] https://www.pjrc.com/tech/8051/ide/fat32.html (12.10.2019.)

[16] https://support.microsoft.com/en-us/help/140365/default-cluster-size-for-ntfs-fat-and-exfat (9.2.2020.)

[17] http://averstak.tripod.com/fatdox/fatintro.htm (12.10.2019.)

[18] http://www.campus64.com/digital_learning/data/cyber_forensics_essentials/info_fat32-ntfs.pdf (17.10.2019.)

[19] https://www.scribd.com/document/336110945/The-Structure-of-NTFS (8.2.2020.)

[20] http://www.ntfs.com/ntfs_basics.htm (9.2.2020.)

[21] https://en.wikipedia.org/wiki/B-tree (10.2.2020.)

[22] https://techdifferences.com/difference-between-fat32-and-ntfs.html (4.3.2020.)

[23] https://en.wikipedia.org/wiki/Comparison_of_file_systems#OS_support (4.3.2020.)

[24] https://en.wikipedia.org/wiki/NTFS (8.4.2020.)

[25] https://www.howtogeek.com/73178/what-file-system-should-i-use-for-my-usb-drive/ (8.4.2020.)

SAŽETAK

Datoteke i direktoriji čine temelj datotečnih sustava. Datoteke predstavljaju smislene cjeline podataka dok direktoriji služe za njihovu dodatnu organizaciju. Datotečni sustav se brine o tome da svi podatci budu smješteni u sekundarnu memoriju na odgovarajući način te nam omogućuje kasnije dohvaćanje tih istih podataka. Ranije je FAT32 bio zadani datotečni sustav Windows računala, ali je s napretkom tehnologije počeo imati previše ograničenja. Zbog toga nije više bio pogodan za računala, ali se i dalje koristi na USB memoriji. Kao zamjena za FAT32 sustav nastao je NTFS koji je uveo znatna poboljšanja te je trenutno zadani datotečni sustav na Windows računalima. Ovdje je objašnjen njihov princip rada te njihove prednosti i nedostatci.

Ključne riječi: datoteka, direktorij, datotečni sustav, FAT32, NTFS

ABSTRACT

Files and directories form the foundation of file systems. Files represent meaningful units of data while directories serve as their additional organization. The file system makes sure that all the data is placed in the secondary memory appropriately and allows us to retrieve that same data later. Previously, FAT32 was the default Windows PC file system, but with the advancement of technology it began to show too many limitations. As a result, it was no longer suitable for computers, but is still used on USB memory. NTFS was created as a replacement for the FAT32 system, which has made significant improvements and is currently the default file system on Windows computers. Their principle of work and their advantages and disadvantages are explained here.

Keywords: file, directory, file system, FAT32, NTFS

ŽIVOTOPIS

Katarina Draganjac rođena je 16. svibnja 1997. godine u Slavonskom Brodu. Osnovnu školu Ivana Kozarca Županja završava 2012. godine kada upisuje Obrtničko-industrijsku školu Županja, smjer Ekonomist. Od 2016. godine studentica je preddiplomskog stručnog studija Elektrotehnike, smjer Informatika na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek. Od kraja 2019. godine pripravnica je za poziciju QA inženjer u tvrtki Bamboo Lab u Osijeku.

_____

Katarina Draganjac