

Kontroler multimedijske pomnilniške enote

Matevž Pogačnik, prof.dr.Jurij Tasič
Fakulteta za elektrotehniko
Univerza v Ljubljani
Tržaška 25, 1001 Ljubljana, Slovenija
matevz.pogacnik@fe.uni-lj.si

Multimedia storage unit controller

Remote Education Application (REA) is using big number of large files, which are stored on 16 Gb linear tape. Due to enormous access time of linear tape, special program is used which copies files from tape to disk and in this way access time is drastically reduced. This program is called the controller. In this paper is described REA, functioning and structure of the controller and the approach to copying of files in order to reduce the access time.

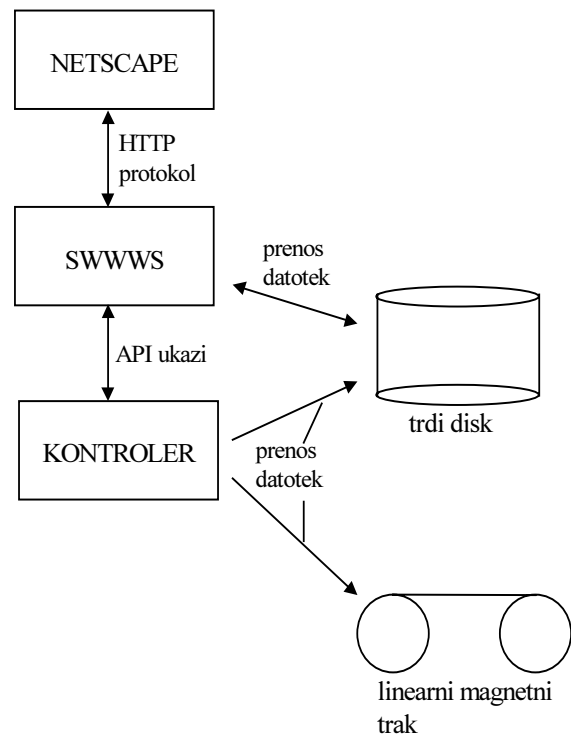
1 Uvod

Razvoj računalniške tehnologije danes omogoča uporabo vse zanimivejših in pestrejših aplikacij. Te vključujejo zvočne in slikovne prikaze, video posnetke, interaktivnost ter druge uporabniku prijazne funkcije. Posledica teh funkcij je uporaba vse večjih datotek, katerih skupna velikost lahko pri posamezni aplikaciji že preseže velikost trdega diska in se povzpne vse tja do nekaj deset Gb in več. Ena takih aplikacij je aplikacija izobraževanja na daljavo, ki jo razvijamo na Fakulteti za elektrotehniko v Ljubljani. Aplikacija med drugim omogoča prikazovanje tekstovnih dokumentov, slik, tabel in video posnetkov z zvočnimi zapisi. Za shranjevanje datotek uporablja izmenljive linearne magnetne trakove s kapaciteto 16 Gb, katerih šibka stran je hitrost dostopa. Povprečni dostopni čas do posameznih datotek na traku znaša dve minuti, kar je nesprejemljivo dolgo, zato je del trdega diska uporabljen kot vmesni pomnilnik. S kopiranjem datotek s traku na disk zmanjšamo dostopni čas, če je večina datotek na disk skopirana vnaprej, torej preden jih aplikacija zahteva. To delo opravlja poseben program imenovan kontroler, ki je natančneje opisan v nadaljevanju.

2 Aplikacija izobraževanja na daljavo

Namen aplikacije je izobraževanje s pomočjo računalnika doma ali na delovnem mestu. Učne lekcije

si lahko predstavljamo kot knjigo, ki poleg teksta in slik vsebuje tudi zvočne zapise, animacije, video posnetke itd. Uporabnik (učenc) se po učni snovi premika z izbiranjem posameznih poglavij, podpoglavij in končno posameznih učnih dokumentov. Ti dokumenti so shranjeni na traku v obliki tekstovnih, zvočnih, video ali drugih datotek in jih je za prikaz v brskalniku potrebno prenesti na trdi disk. Za lažje razumevanje delovanja sistema je njegova arhitektura predstavljena na sliki 1.



Slika 1: Arhitektura sistema

2.1 Zgradba aplikacije

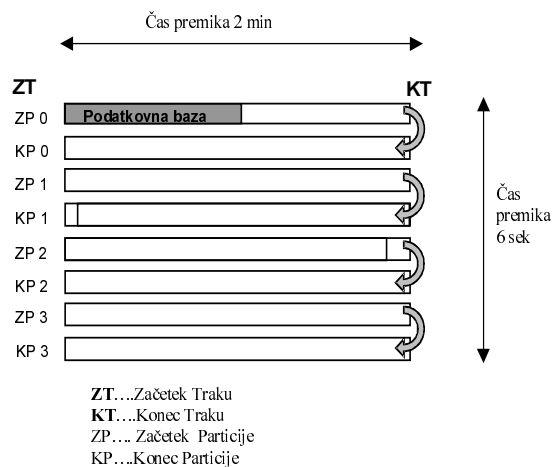
Aplikacija sama je zgrajena iz javanskih programčkov imenovanih applet-i, ki se izvajajo v brskalniku¹. Na podlagi izbranega učnega dokumenta brskalnik od preprostega spletnega strežnika (SWWWS - Simple WWW Server) zahteva ustrezne datoteke. SWWWS poskuša zahtevane datoteke najti na trdem disku in jih pošlata brskalniku, glej sliko 1. V primeru da zahtevanih datotek ni na trdem disku, SWWWS od kontrolerja zahteva prenos teh datotek s traku. Ko kontroler javi uspešen prenos datotek na disk, jih SWWWS prenese brskalniku. Komunikacija in prenos datotek med SWWWS-em in brskalnikom poteka preko HTTP protokola, med SWWWS-em in kontrolerjem pa preko posebnih API (Application Protocol Interface) klicev.

Zaradi nekaterih lastnosti uporabljenega linearnega traku, kot so relativna počasnost in nenaključnost dostopa do posameznih datotek, je delovanje aplikacije do neke mere oteženo oz. upočasnjeno. Delovanje kontrolerja, ki kopira datoteke s traku na disk, je zato prirejeno tem lastnostim. Za lažje razumevanje delovanja kontrolerja so lastnosti linearnega traku natančneje opisane.

2.2 Linearni trak

Linearni trak uporabljen za aplikacijo izobraževanja na daljavo je Tandbergov magnetni trak (MLR1). Na računalnik je priključen preko SCSI vmesnika, v primeru aplikacije izobraževanja na daljavo pa je uporabljan na operacijskem sistemu Linux. Pomnilniška kapaciteta posamezne kasete je 13 do 25 Gb, trenutno pa je uporabljana 16 Gb izvedba. Trak je razdeljen na 36 t.i. particij z dolžino 420 Mb [2,3]. Kot je razvidno iz slike 2 se vsaka particija začne na enem koncu traku, teče do drugega konca, se tam obrne in vrne na začetni konec traku. To pomeni da imajo particije svoj začetek in konec na istem delu traku. Čas premikanja traku od enega konca do drugega v vzdolžni smeri je 2 minuti, posledično je čas premika od enega do drugega konca particije po vsej njeni dolžini 4 minute. Prečni premik bralno-pisalne glave traja približno 6 sekund, kar ustreza času premika med začetkom prve in koncem zadnje particije. Uporabljeni linearni trak ima tudi to lastnost da je hitrost premikanja traku neodvisna od aktivnosti bralno-pisalne glave, torej je enaka ne glede na to ali ta bere s traku, nanj zapisuje ali pa se trak samo vrti [2,3]. To pomeni da traja dostop do posamezne datoteke enako dolgo kakor kopiranje vseh datotek, ki so pred njo na isti particiji.

¹ V tem trenutku je za ta namen najprimernejši brskalnik Netscape 4.05, ki ima najboljšo podporo za izvajanje javanskih programov verzije 1.1.



Slika 2: Zgradba linearnega traku

2.3 Kontroler

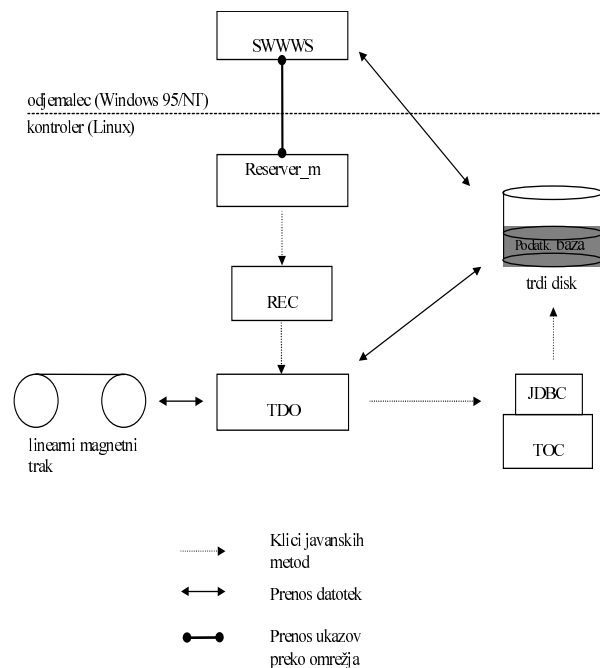
Kontroler aplikacije je program, katerega osnovna funkcionalnost je kopiranje datotek s traku na disk in v obratni smeri. Napisan je v programskem jeziku Java. Eden izmed razlogov, da je bil za kontroler izbran prav ta programski jezik, je v tem, da je Java objektno orientiran programski jezik. Programiranje v objektno orientiranih jezikih omogoča modularno zasnovno programov, kar pomeni da lahko tem programom kasneje dodajamo nove funkcionalnosti in brez večjih posegov spreminjamo obstoječe. Ob pravilni zasnovi programa z dodajanjem novih modulov sestavljamo novo, več-funkcionalno celoto. Poleg tega ima Java že pripravljena orodja (razrede) za delovanje programov preko omrežij, branje in pisanje iz/v datoteke in ustvarjanje grafičnih vmesnikov, če naštejemo le nekatere. Uporaba vseh teh orodij je zelo preprosta saj programerja razbremeni podrobnosti izvajanja programov kot so naslavljanje ter sproščanje pomnilnika, uporaba kazalcev in podobno. Dobra stran Jave je med drugim tudi večopravnost (večnitnost), ki omogoča navidezno hkratno izvajanje več procesov [1]. S tem pospešimo izvajanje programov, kar je zelo dobrodošlo, saj Java v tem trenutku ni prav "hiter" jezik. Problem hkratnega (nenadzorovanega) dostopa do posameznih sredstev (npr. vhodno-izhodnih naprav, datotek...), ki lahko nastopi kot posledica večopravnosti, je rešen z uvedbo sinhroniziranih metod [1]. Nenazadnje je dobra stran Jave v tem, da lahko katerokoli javansko kodo izvajamo na poljubni podlagi oziroma računalniku. Za to potrebujemo le ustrezen javanski prevajalnik (Java Virtual Machine – JVM), ki pa so na voljo za praktično vse glavne računalniške sisteme.

Java ima tudi nekaj slabih lastnosti. Ena od njih je že omenjena hitrost izvajanja, ki zaenkrat še močno zaostaja za drugimi programskimi jeziki. Razvijalci Jave že povečujejo hitrosti javanskih prevajalnikov, tako da lahko v kratkem pričakujemo izboljšave tudi na tem področju. V primeru aplikacije izobraževanja na daljavo hitrost izvajanja javanskih programov ni povzročala večjih težav in je presegala hitrost najpočasnejših komponent sistema. Naslednji problem se je pokazal pri izvajanju sistemskih klicev. Java sicer ima mehanizme za njihovo izvajanje, vendar so orodja za preverjanje njihove uspešnosti zaenkrat še precej okorna. Ker zaenkrat še ni mogoče dosledno zasledovati pojavljanja napak pri izvajanju sistemskih klicev, smo v primeru naše aplikacije uvedli nekaj zasilnih mehanizmov, s katerimi rešujemo ta problem.

2.3.1 Zgradba kontrolerja

Kontroler je večnivojski javanski program, pri katerem vsak ima vsak od nivojev svojo funkcijo. Zgrajen je iz štirih glavnih in nekaj pomožnih razredov. Glavni razredi so razporejeni po funkcionalnih nivojih, glej sliko 3.

Na najvišjem nivoju je razred REserver_m (Remote Education server multithreaded), katerega naloga je komunikacija s SWWWS-jem in sprejemanje ukazov od njega, glej sliko 1. Zaradi svoje večnitosti lahko razred Reserver_m, s pomočjo dveh pomožnih razredov, hkrati sprejme poljubno število ukazov. Če prepozna ukaz in ukazu sledi ustrezno število parametrov, ga posreduje razredu na naslednjem nivoju, to je razredu REC (Remote Education Controller). Ta preveri vrednosti parametrov, razpoložljivost sredstev (zasedenost traku, prostor na disku) in pripravi vse potrebno za izvajanje ukaza (sezname datotek, če je potrebno naredi prostor na disku itd.). Ko je vse pripravljeno za izvajanje ukaza ga REC posreduje razredu na naslednjem nivoju, razredu TDO (Tape Disk Operator). Po preverjanju obeh predhodnih razredov ta dejansko izvrši ukaze oz. izvede kopiranje s traku na disk ali v obratni smeri. Zadnji od glavnih štirih razredov je razred TOC (Table Of Contents). Ta deluje kot povezava z bazo podatkov, v kateri so shranjeni za aplikacijo potrebni podatki o datotekah na traku. Glavni podatki v njej so imena datotek, njihove velikosti, particija(e) kateri(m) pripada(jo) poleg tega pa tudi podatki o samih particijah, kot so število in velikost datotek na posamezni particiji in drugi. Razred TOC je z bazo podatkov povezan preko JDBC (Java Data Base Connectivity) gonilnikov. Do podatkov v bazi dostopa preko SQL ukazov.



Slika 3: Zgradba kontrolerja

2.3.2 Kontrolerjevi ukazi

Glavni kontrolerjevi ukazi so nabor API klicev, ki omogočajo izvajanje aplikacije neodvisno od pomnilniškega medija. Sama aplikacija teh ukazov ne pozna, saj se izmenjujejo le med SWWWS-jem in kontrolerjem. Definirani so bili v sodelovanju s skupino iz Philipsa, ki se ukvarja s sorodnim sistemom in katerega del predstavlja aplikacija izobraževanja na daljavo. Nekaj ukazov je namenjenih začetku in koncu dela z aplikacijo, to so ukazi *Insert*, *Open session* in *Close Session*. Prvi je uporabljen ob začetku dela s praznim trakom, drugi ob začetku dela s trakom, ki vsebuje datoteke in zadnji ob zaključku dela s poljubnim trakom. Ob klicu prvih dveh metod se ustvari nova podatkovna baza oz. se obstoječa skopira s traku na disk.

Ukaz, ki s traku skopira vsebino posamezne particije se imenuje *Restore*, z ukazom *Archive* pa poženemo kopiranje datotek z diska na trak. Poizvedovanje po številki particije posamezne datoteke izvede ukaz *LookUp*, seznam vseh datotek na traku izpiše ukaz *List* in velikost nezasedenega prostora na disku nam vrne ukaz *Free*. Nekateri izmed naštetih ukazov imajo več različic², vendar to za razumevanje delovanja kontrolerja ni bistveno.

² Npr. z ukazom *Free* lahko poizvedujemo po velikosti prostega prostora na celotnem traku ali pa na posamezni particiji.

2.3.3 Problem kopiranja datotek s traku

Glavni problem kopiranja datotek s traku na disk, je povprečni dostopni čas do naključno razporejenih datotek. Ta znaša pribl. 2 minuti, kar je za današnjega uporabnika povsem nesprejemljivo, zato ga skušamo zmanjšati z različnimi metodami.

Zaradi enake hitrosti premikanja traku pri branju/pisanju in samem vrtenju traku, kontroler vsako kopiranje datotek začne na začetku particije, na kateri se nahaja zahtevana datoteka. Na disk tako kopira tudi datoteke, ki se nahajajo pred zahtevano datoteko. Ko je le-ta skopirana na trak, kontroler kopiranja samodejno ne prekine, temveč nadaljuje s kopiranjem preostalih datotek na tej particiji. Ob sprejemu ukaza po kopiranju nove datoteke najprej preveri, če je zahtevana datoteka morda na particiji, ki se trenutno kopira na disk. Če je, kopiranja ne prekine, v nasprotnem primeru pa začne s kopiranjem particije na kateri se nahaja zahtevana datoteka.. Da bi dosegli čim večjo učinkovitost takega pristopa je potrebno datoteke, za katere je verjetnost, da bodo zahtevane hkrati oz. v istem kontekstu, posneti na isto particijo³. Tako ob kopiranju zahtevane datoteke na disk kopiramo tudi datoteke, za katere verjamemo da bodo v kratkem zahtevane s strani aplikacije in tako skrajšamo povprečni dostopni čas do posameznih datotek.

Drugi problem nastopi, ko se disk napolni z datotekami in kontroler dobi zahtevo po kopiranju novih datotek s traku. Potrebno se je odločiti katere datoteke obdržati na disku in katerih ne. Na disku je namreč le 1 do 2 Gb prostora, skupna dolžina datotek na traku pa je lahko do 16 Gb. V tem primeru se kontroler odloči za katero particijo (njene datoteke) je v tistem trenutku najmanjša verjetnost, da jo bo aplikacija v kratkem potrebovala. Nato pobriše vse datoteke, ki pripadajo tej particiji. To ponavlja dokler na disku ni dovolj prostora za particijo, katere datoteka je bila zahtevana. Algoritem odločanja o tem katero particijo pobrisati z diska je eden od modulov kontrolerja, zato ga lahko brez težav nadomestimo z drugim, boljšim.

3 Zaključek

Aplikacija izobraževanja na daljavo za svoje delovanje potrebuje velike količine podatkov oz datotek.

Trenutno je za njihovo shranjevanje uporabljen linearni magnetni trak z veliko kapaciteto, ki pa ima zelo slab povprečni dostopni čas do posameznih datotek. Zato aplikacija datoteke jemlje neposredno z diska, kontroler pa jih kopira s traku na disk, pri čemer poskuša uporabljati čim boljšo strategijo kopiranja in

brisanja datotek. Zaradi njegove modularne zasnove je mogoče posamezne funkcionalnosti izboljševati in dopolnjevati, uporaba programskega jezika Java pa omogoča uporabo tega kontrolerja na različnih računalnikih.

Literatura

- [1] Daniel J. Berg and J. Steven Fritzing, *Advanced Techniques for Java Developers*, Wiley Computer Publishing, 1998
- [2] Ivar Mileteig and Edwin Montie, *Submission of a Multimedia File Format for Linear Recording*, SMASH Deliverable #10, 1997
- [3] Tandberg Data Company, *Tandberg MLR1 Series*, Reference Manual, 1996

³ Za optimalno razvrščanje datotek na trak smo v okviru delovne skupine razvili poseben optimizacijski program, ki med drugim upošteva pripadnost datotek posameznim poglavjem učne snovi.

