

Distribuirani FS (Sistemi Datoteka)

- Pozadina
- Imenovanje i Transparentnost
- Udaljni Pristup datotekama
- **Statefull v Stateless Services**
- File Replikacije
- Primeri Sistema

Pozadina

- **Distribuirani FS (DFS):**
 - ☞ distribuirana implementacija
 - ☞ klasičnog time-sharing modela FS (file system),
 - ☞ gde više korisnika dele datoteke i memorijske resurse
- **DFS upravlja skupom udaljenih memorijskih uređaja (diskova)**
- **Ukupan memorijski-disk prostor sa kojim upravlja DFS**
 - ☞ je sastavljen od različitih
 - ☞ udaljenih
 - ☞ manjih memorijskih prostora (diskova i mašina)
- **Obično postoji korespondencija između:**
 - ☞ konzistentnog disk-memorijskog prostora i
 - ☞ skupa datoteka.

Struktura DFS-a

- **Servis:**
- **softverska celina tj softver**
 - ☞ Koji se koristi na jednoj ili više mašina
 - ☞ obavlja određene funkcije
 - ☞ da obezbedi uslugu nepoznatim klijentima
- **Server:**
- **softver koji se koristi za servis se nalazi na jednoj mašini.**
- **Client:**
- **proces koji može da pozove servis**
 - ☞ korišćenjem skupa operacija
 - ☞ koje formulišu njegov klijentski interfejs.
- **Klijentski interfejs za FS**
 - ☞ je formiran od
 - ☞ seta primitivnih FS operacija (create, delete, read, write).
- **Klijentski interfejs DFS-a**
 - ☞ treba da bude **transparentan**,
 - ☞ ne sme da bude različit između lokalnih i udaljenih datoteka

Imenovanje i Transparentnost

- **Imenovanje:** mapiranje između logičkih i fizičkih objekata
- **Multilevel mapping:**
 - ☞ apstrakcija datoteke
 - ☞ koja **krije detalje** o tome kako i i gde su
 - ☞ na disku datoteke smeštene

FS:/directory path/filename

hostname:FS:/directory path/filename

- **Transparentni DFS**
 - ☞ krije lokaciju
 - ☞ gde unutar mreže je datoteka smeštena.
- **Za datoteke koji su iskopirane na više sajtova,**
 - ☞ mapiranje vraća set lokacija kopija te datoteke;
 - 📄 kopije i njihove lokacije
 - 📄 su sakrivene.

Imenovanje Struktura

■ Transparentnost lokacija:

- ☞ ime datoteke **ne pokazuje**
📄 **fizičku lokaciju** te datoteke

■ Karakteristike

- ☞ Ime datoteke i dalje označava **specifičan**, mada skriven,
📄 **set fizičkih disk blokova**
- ☞ **Pogodan način za deljenje podataka**
- ☞ **Može da otkrije korespondentnost između**
📄 **komponenti i mašina.**

■ Nezavisnost lokacija:

- ☞ **ime datoteke ne treba da se menja**
- ☞ **kada se promeni njena fizička lokacija.**

■ Karakteristike

- ☞ **Bolje izdavanje datoteka**
- ☞ **Unapređuje deljenje memorijskog prostora.**
- ☞ **Odvaja hierarhiju imenovanja**
📄 **od memorijske hijerarhije**

Šeme Imenovanja — Tri Osnovne Metode

■ I Jedinstven system-wide name

- ☞ datoteke dobijaju imena
- ☞ kombinacijom njihovog host imena i lokalnog imena;
- ☞ Garantuje jedinstveno ime unutar sistema.

■ II Dodaje udaljene direktorijume lokalnim direktorijumima

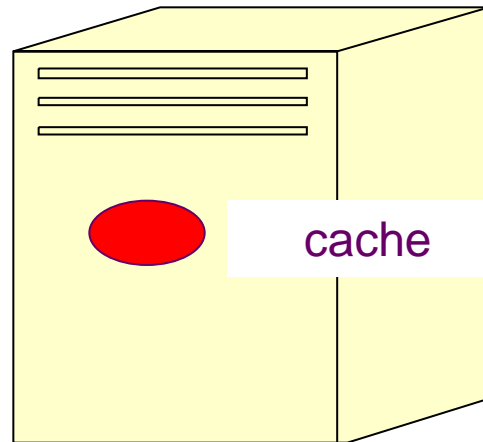
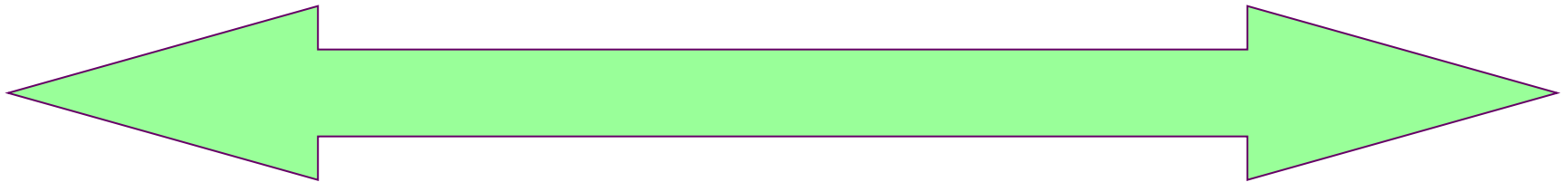
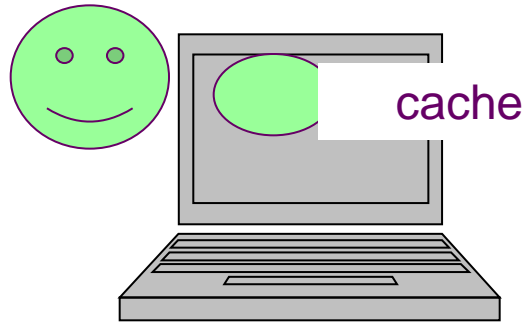
- ☞ daje izgled jedinstvenog stabla direktorijuma;
- ☞ samo prethodno mount-ovanim udaljenim direktorijumima
 - 📁 može transparentno da se pristupi

■ III Sveobuhvatna integracija komponenti FS (sistema datoteka)

- ☞ Jedinstveno globalno ime strukture obuhvata sve datoteke u sistemu.
- ☞ ako server nije dostupan,
 - 📁 neki proizvoljan set direktorijuma na različitim mašinama
 - 📁 takođe postaje nedostupan

Pristup udaljenim datotekama

- 1. remote access
- 2. using cache



master copy

Pristup Udaljenim datotekema

■ 1. Remote Service

- ☞ koristi rpc pozive
- ☞ za upis ili čitanje
- ☞ dela datoteke koji je klijent dobio preko mreže

■ 2. Keširanje za globalno poboljšanje performansi (ali...)

- ☞ ako potrebnii podatak već nije keširan
 - 📄 kopija podatka se dobija od servera do korisnika
 - 📄 uvek se pristupa keširanoj kopiji
- ☞ datoteke se indentifikuju sa jednom master kopijom
 - 📄 MC je smeštena na jednom serveru,
 - 📄 ali
 - 📄 kopije(delovi) su rasuti po različitim mestima.
- ☞ **Cache-consistency problem**
 - 📄 čuvanje keširanih kopija je koizistentno master datoteci.

Lokacija Keša – Disk vs. Glavne Memorije

time to data = network time + local disk time access

■ Prednosti disk keša (DCD)

- ☞ **Mnogo pouzdaniji**
- ☞ Keširani podaci se čuvaju na disku
- ☞ **su i dalje tu za vreme oporavka**
- ☞ **nije potrebno da se ponovo dovlače (fetch)**

■ Prednosti memorijskog keša:

- ☞ **Dozvoljava da radne stanice budu bez diska**
- ☞ **Brži pristup podacima**
- ☞ **Veća memorija i može da se izvodi ubrzanje.**
- ☞ **Serverski keš** (koji se koriste da ubrzaju disk I/O) je u glavnoj memoriji
 - 📄 **bez obzira gde je korisnički keš lociran;**
 - 📄 korišćenjem memorijskog keša na korisničkoj mašini se omogućava
 - 📄 jedinstven keš mehanizam za servere i korisnike

Cache Update Policy

■ Write-through: upis kroz

- ☞ Upisivanje podataka kroz disk
- ☞ umesto u bilo koji keš.
- ☞ Pouzdano, ali slabijih performansi

■ Delayed-write – odloženi upis

- ☞ modifikacije se upisuju u keš
- ☞ a zatim na server.

☞ Brži pristup za upis;

- 📄 neki podaci mogu biti prekucani pre nego što se upišu ponovo
- 📄 tako da oni uopšte ne treba da se upisuju.

☞ nepouzdan;

- 📄 neupisani podatak će se izgubiti kad god se mašina blokira.

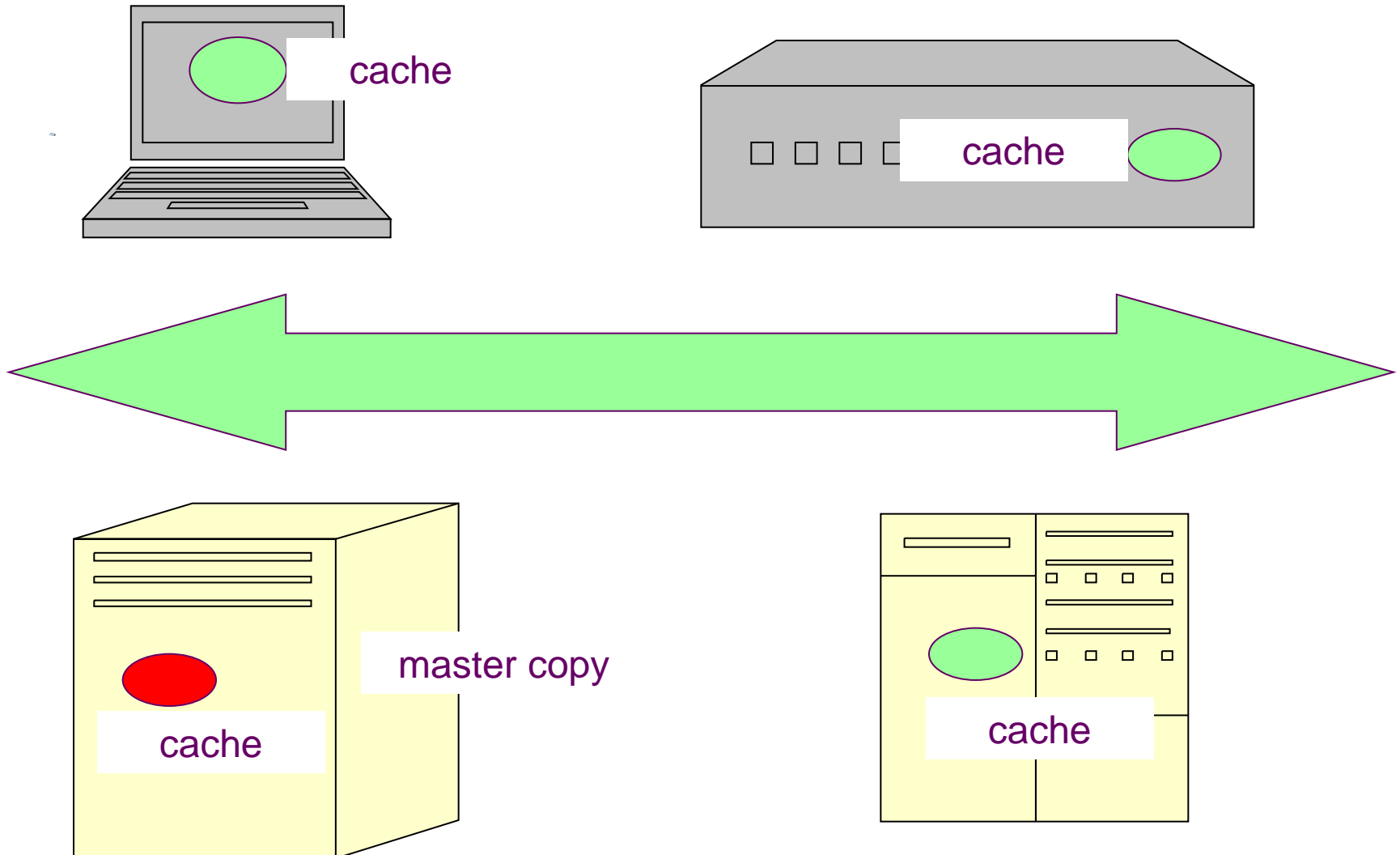
☞ Variation – flushing

- 📄 keš se skenira u regularnim intervalima
- 📄 prazni blokove koji su bili modifikovani nakon poslednjeg skeniranja

☞ Variation – write-on-close,

- 📄 upisuje blokove podataka na server
- 📄 kada se datoteka zatvara.
- 📄 najbolje za podatke koji su otvoreni na duže vreme i često se menjaju

Konzistencija



Konzistencija

- Da li je **lokalno keširana kopija**
- **podatka konzistentna master kopiji?**
- **Pristup iniciran od strane klijenta (Client-initiated approach)**
 - ☞ **Klijent započinje proveru vrednosti**
 - ☞ **Server proverava da li**
 - 📄 **su lokalni podaci konzistentni**
 - 📄 **master kopiji**
- **Pristup iniciran od strane servera (Server-initiated approach)**
- **Server vodi zapis,**
 - 📄 **o svakom klijentu,**
 - 📄 **o (delovima datoteka) datotekema koje dobije**
 - ☞ **Kada server detektuje potencijalnu nekonzistenciju, mora da reaguje**

Poređenje Keširanja i Udaljenog Servisa

- **Kod keširanja,**
 - ☞ **lokalni keš efikasno rukovodi većinom udaljenih pristupa;**
 - ☞ većina udaljenih pristupa se opsluđuje istom brzinom kao i lokalni.
- **Kod keširanja serverima se pristupa samo s vremena na vreme**
 - ☞ (pre nego za svaki upis).
 - ☞ Smanjuju serverski i mrežni saobraćaj
 - ☞ Povećava potencijal za scalability
- **Metod udaljenog servera**
 - ☞ rukovodi svim udaljenim pristupima preko mreže;
 - ☞ o obara performanse zbog mrežnog saobraćaju, čitanje/upis sa servera
- **Total network overhead**
 - ☞ **in transmitting big chunks of data (caching)**
 - ☞ **is lower than**
 - ☞ **a series of responses to specific requests (remote-service)**

Keširanje i Udaljeni Servisi (Nastavak)

■ Keširanje je najbolje

- ☞ kod pristupa delovima sa **ređim upisima (read dominant)**.
- ☞ kod **čestih upisa**,
 - 📄 su bitni opšti troškovi
 - 📄 da bi se prevazišao problem konzistencije keša.

■ Keširanje je dobro za mašine sa:

- ☞ lokalnim diskovima
- ☞ velikim RAM memorijama.

■ Udaljeni pristup je dobar za:

- ☞ mašine bez diska,
- ☞ sa malim memorijskim RAM kapacitetom

Stateful File Services

■ **Mehanizam:**

- ☞ Klijent otvara datoteku.
- ☞ Server fečuje informaciju o datoteci sa svog diska,
 - 📄 smešta je na svoju memoriju
 - 📄 daje klijentu **identifikator konekcije**
 - 📄 jedinstven za klijenta i otvara datoteku.
- ☞ **Identifikator** se koristi za sledeće pristupe dok se ne sesija ne završi .
- ☞ Server mora da podesi informaciju u glavnoj memoriji
 - 📄 koju je koristio klijent
 - 📄 koji više nije aktivan

■ **Povećane performanse:**

- ☞ **Smanjeni pristup disku**
- ☞ **Stateful server zna:**
 - 📄 da li je datoteka bila otvorena za sekvencijalni pristup
 - 📄 pa može da čita unapred sledeće blokove.

Stateless File-Server

■ Izbjegava informacije o stanju

- pravljenjem
- nezavisnih zahteva

■ Svaki zahtev

- identifikuje datoteku
- i
- poziciju u datoteci

■ Nema potrebe za uspostavljanje i raskidanje veze

- operacijama open i close

Razlika Između Stateful & Stateless Servisa

■ Oporavak od otkaza:

- ☞ Stateful server gubi sve svoje memorijske informacije usled pada
 - 📄 obnavlja stanje
 - 📄 preko **recovery protokola**, zasnovanog na dialogu sa klijentom,
 - 📄 i prekinute operacije
 - 📄 koje su bile u toku kada se dogodio pad

- ☞ **Server treba da bude svestan otkaza od strane klijenta**
 - 📄 zbog zahteva za popravljjanje prostora
 - 📄 dodeljenog za upis mesta oktazalih klijent procesa
 - 📄 (detekcija i eliminacija siročića).

- ☞ **Kod stateless servera,**
 - 📄 oporavak je skoro neprimetan.

- ☞ **Oporavljeni server**
 - 📄 može da odgovori
 - 📄 nezavisnim zahtevima bez ikakvih poteškoća

Razlike (Nastavak)

■ Nedostaci korišćenje snažnih stateless servisa:

- ☞ duže request poruke
- ☞ sporije procesiranje zahteva
- ☞ dodatna ograničenja nametnuta na dizajn DFS

■ Neki okruženja zahtevaju stateful servise:

- ☞ Server koji koristi server-initiated cache validation
- ☞ Ne može da sprovodi stateless servis,
 - 📄 pošto sadrži zapis
 - 📄 koji datoteke su keširani od strane kog klijenta
- ☞ UNIX koristi fajl deskriptore i implicitne ofsete
- ☞ i to je stateful servis;
 - 📄 serveri moraju da sadrže tabele
 - 📄 za mapiranje fajl deskriptora za inode,
 - 📄 i
 - 📄 smeštanje konkretnog ofseta unutar datoteke

Replikacija datoteka

- **Replikacije iste datoteke**
 - ☞ smeštene na failure-independent mašinama
- **Povećava korisnost i može da skрати vreme servisiranja.**
- **Šeme za imenovanje** mapiraju replicirano ime datoteke za određenu repliku
 - ☞ Postojenje replika bi trebalo da bude **nevidljivo** za više nivoa.
 - ☞ Replikacije moraju da se odlikuju
 - ☞ **različitim imenima nižih nivoa**
- **Ažuriranje:**
 - ☞ replike datoteke označavaju isti logički smisao,
 - ☞ i zbog toga
 - ☞ ažuriranje bilo koje replike
 - ☞ mora da se reflektuje **na sve ostale replike.**
- **Demand replication:**
 - ☞ ažuriranje **samo** unutar **primarne replike,**
 - ☞ onda obaveštavanje drugih replikacija o **invalidaciji**