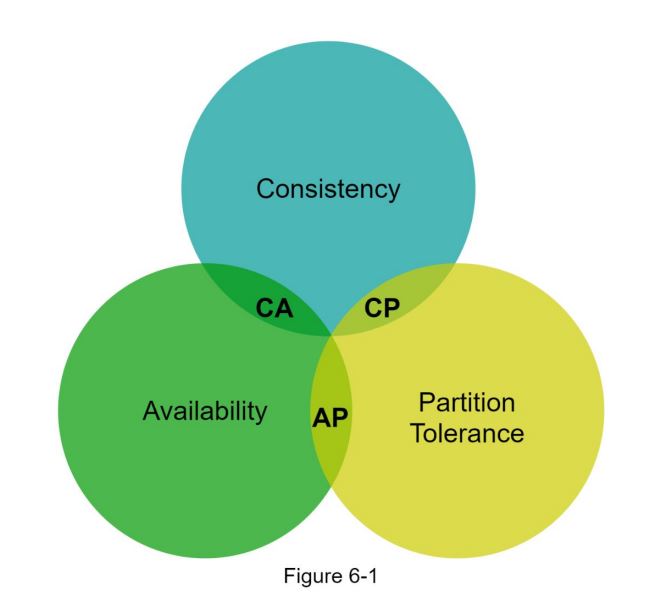
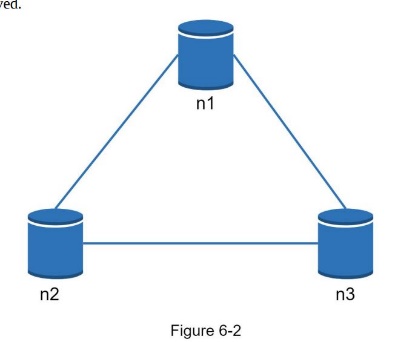
**Key-Value Store i CAP Teorem**

Key-value store je distribuiran sustav za pohranu podataka gdje se svaki zapis sastoji od jedinstvenog ključa i pripadajuće vrijednosti. Osnovne operacije su put(key, value) i get(key).

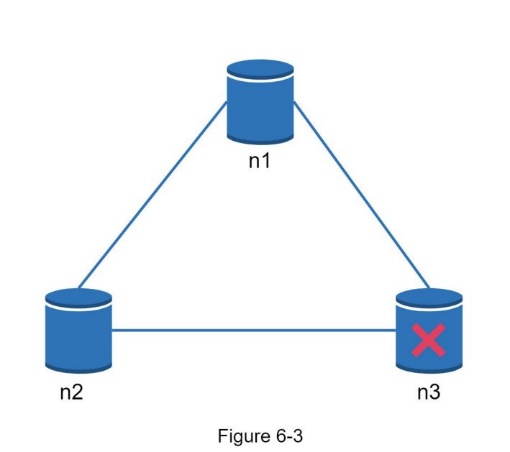


CAP teorem pokazuje da distribuirani sustavi mogu zadovoljiti najviše dva od tri svojstva: dosljednost (C), dostupnost (A) ili toleranciju particija (P).  
sustav može garantirati samo dva od tri svojstva: **C**onsistency, **A**vailability, **P**artition Tolerance.

****

Idealni scenarij sinkronizacije podataka na trima replikama (n1, n2, n3) bez prekida mreže. Svi čvorovi imaju iste podatke, što osigurava dosljednost i dostupnost.

**CP sustavi**   
  
(npr. banke) – prioritiziraju dosljednost, ali mogu odbiti zahtjeve tijekom mrežnih particija (Fig. 6-3).



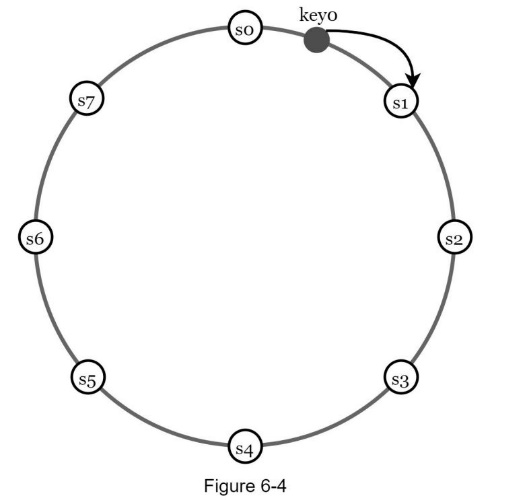
Mrežna particija onemogućuje komunikaciju s n3, što dovodi do nedosljednosti između n1/n2 i n3. Sustav mora birati između dostupnosti (AP) ili dosljednosti (CP).

**Dosljedno hashiranje**

 **Korisnik zatraži pohranu ključa**, npr. "korisnik:42"

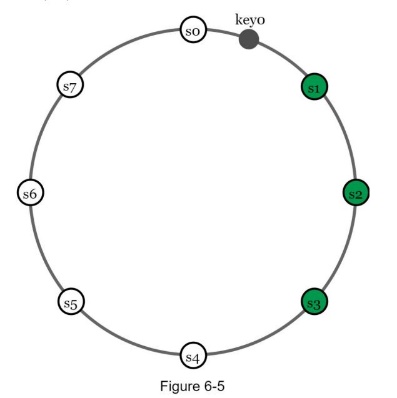
 Sustav taj ključ **hashira** pomoću neke hash funkcije (npr. SHA-1, MD5, CRC32…).

Ravnomjerno raspodjeljuje podatke i omogućuje automatsko skaliranje.



Dosljedno hashiranje raspodjeljuje podatke na 8 poslužitelja (s0-s7) smještenih u prsten. Ključevi se dodjeljuju prvom dostupnom čvoru u smjeru kazaljke na satu.  
  
 Ključ " korisnik:42" hashira se na poziciju na prstenu i ide na najbliži čvor (recimo čvor B).

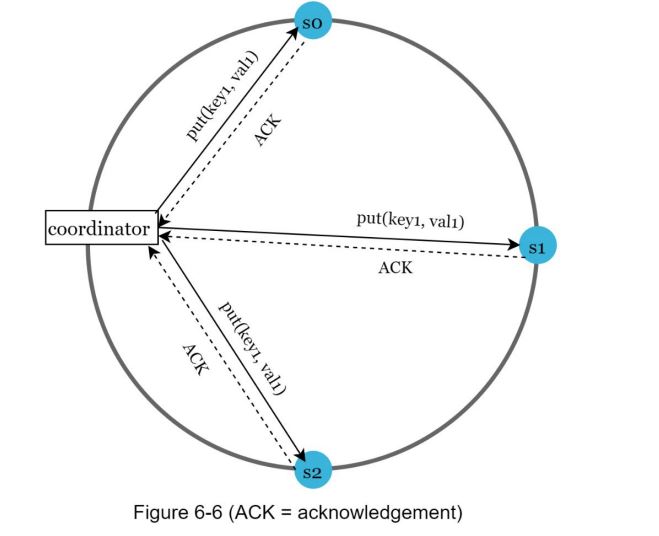
 Ako dodamo novi čvor D, **samo dio ključeva** koji su ranije išli u B sada idu u D — ostali ostaju gdje jesu!

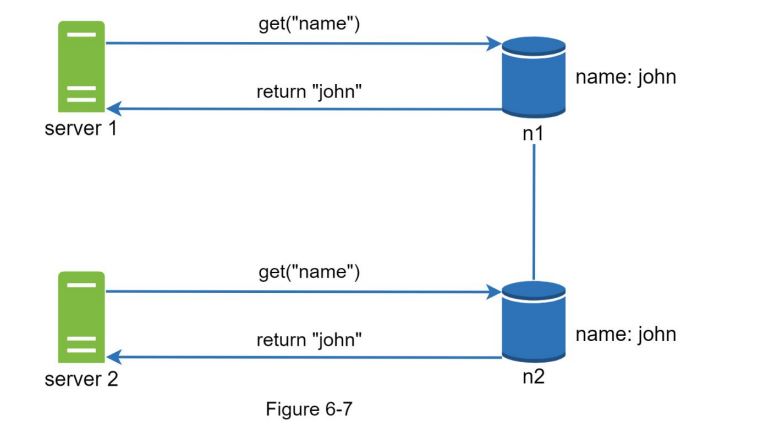


**Replikacija**   
Replikacija podataka znači da se **isti podatak čuva na više čvorova (servera)**, umjesto samo na jednom.

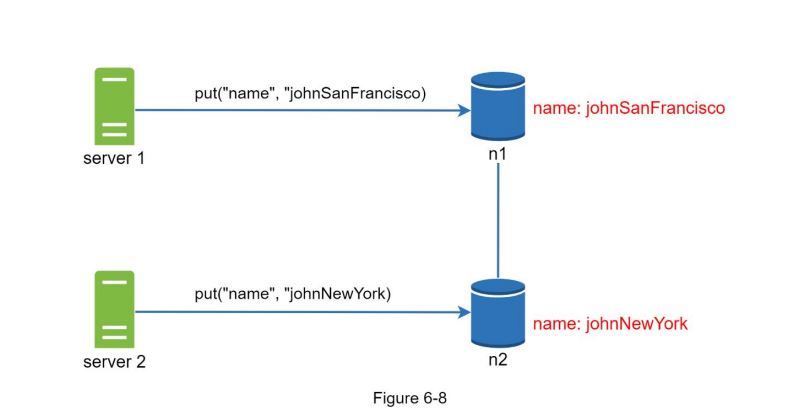
podaci se kopiraju na više čvorova (npr. N=3) radi otpornosti na kvarove

Replikacija podataka (N=3) osigurava otpornost na kvarove. Ključ "key0" se pohranjuje na s1, s2 i s3, što omogućuje oporavak ako jedan čvor otkaže.



Koordinator šalje zahtjev za pisanje i čeka potvrde (ACK) od W replika. Ako W=1, operacija je brza, ali može ugroziti dosljednost.  
  


Oba poslužitelja vraćaju istu vrijednost "john" za ključ "name", što pokazuje uspješnu sinkronizaciju u stabilnom sustavu bez sukoba.  
  
  
  
**AP sustavi** (npr. Dynamo) – prioritiziraju dostupnost, ali mogu vratiti zastarjele podatke (Fig. 6-8).

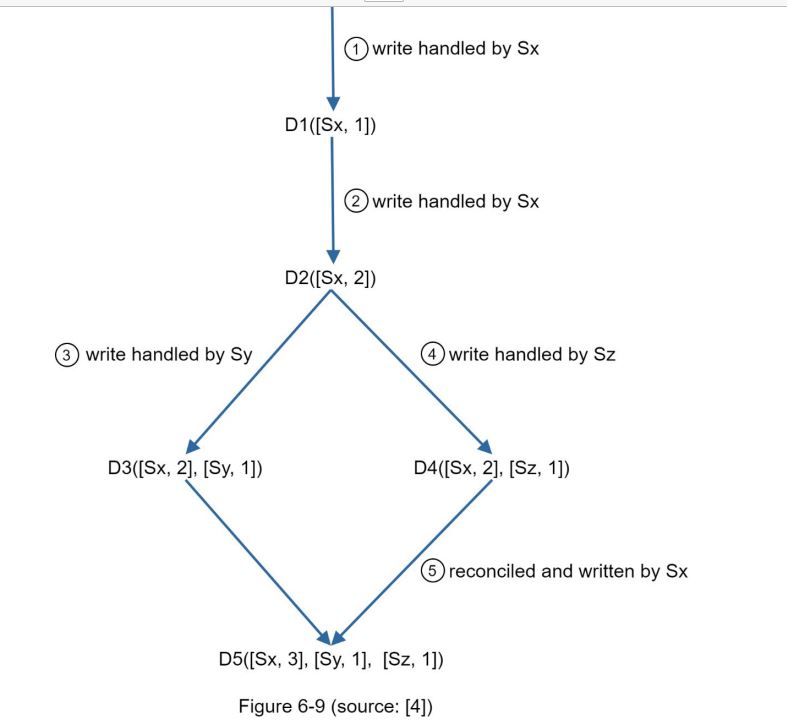


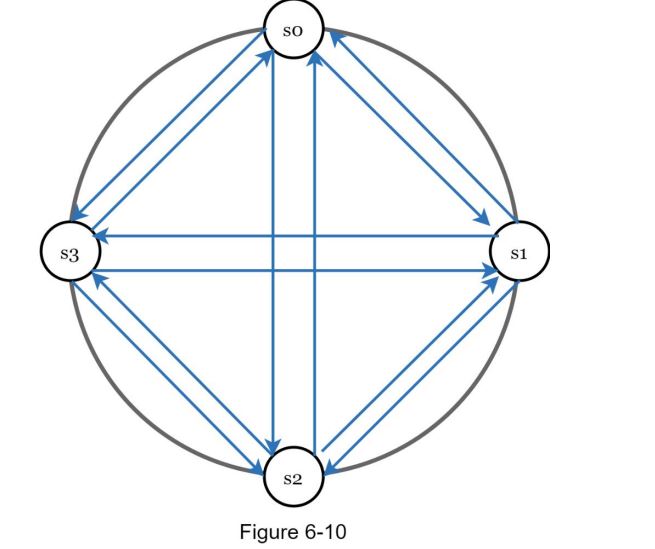
Istovremeno ažuriranje ključa "name" na različitim čvorovima stvara sukobljene verzije (v1 i v2). Rješenje je vektorski satovi ili ručno rješavanje.

**Detekcija i oporavak od kvarova:**

**Vektorski satovi**   
su vrlo važan alat za **praćenje verzija podataka** u **distribuiranim sustavima**, pogotovo kada postoji **asinkrona replikacija** i sustav dopušta **eventualnu konzistentnost** (npr. Dynamo, Riak).

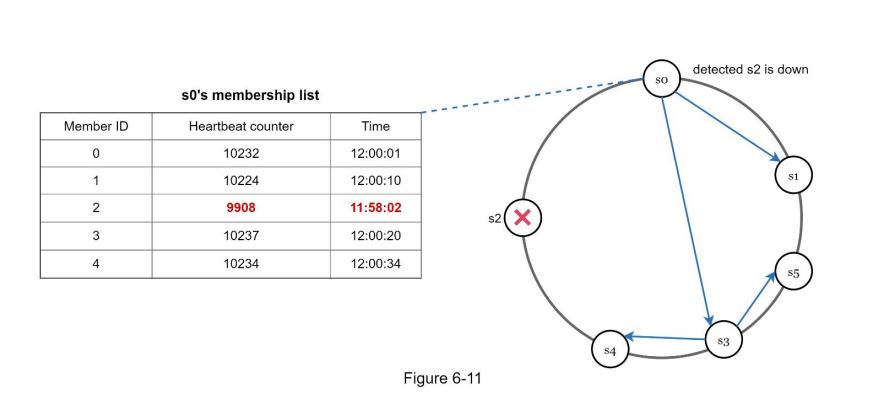
Prate verzije podataka i detektiraju sukobe.



Vektorski satovi prate izmjene podataka na različitim čvorovima (npr. Sx, Sy, Sz). Sukob se detektira ako verzije nemaju jasnu hijerarhiju.  


Detekcija kvarova putem "all-to-all" multicastinga nije skalabilna jer zahtijeva previše resursa u velikim klusterima.

* **Gossip protokol** (Fig. 6-11) – decentralizirano otkriva nedostupne čvorove.
* **Gossip protokol**
* Decentralizirano otkriva nedostupne čvorove.

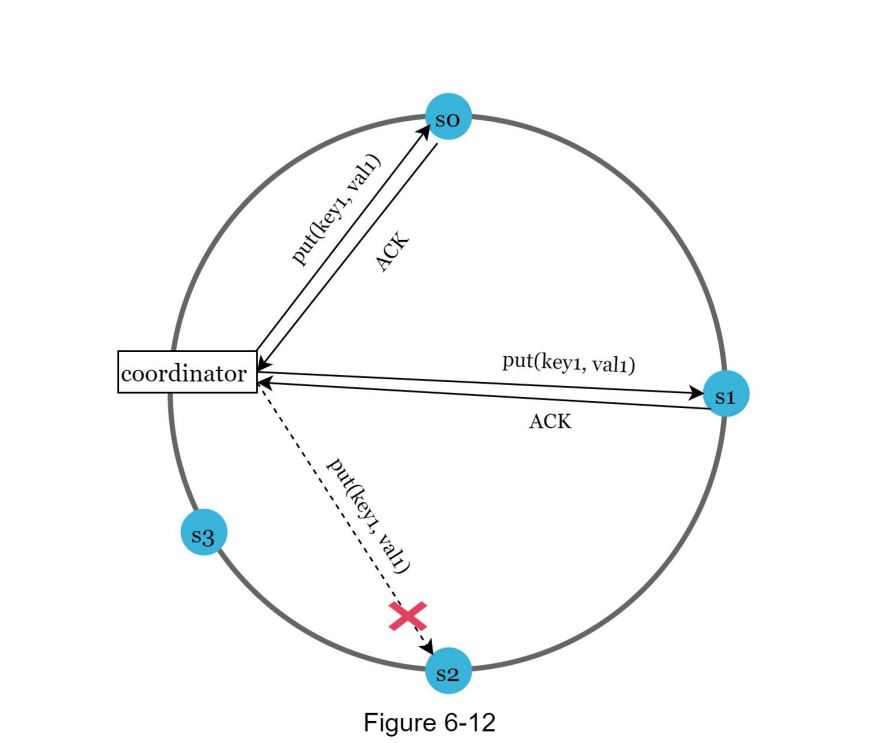


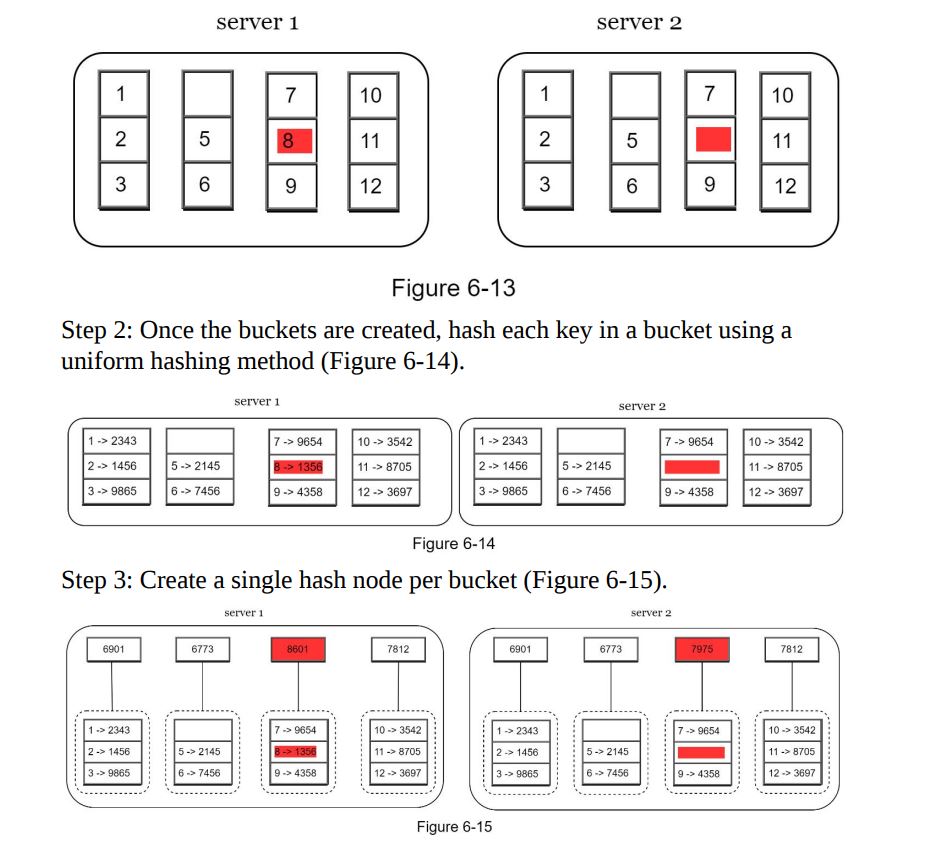
Gossip protokol omogućuje decentraliziranu detekciju kvarova. Čvor s0 primjećuje da s2 nije poslao heartbeat te širi tu informaciju drugim čvorovima.

**Sloppy quorum & hinted handoff**   
  
**Sloppy quorum** i **hinted handoff** su dvije ključne tehnike koje koriste sustavi poput **Amazon Dynamo**, **Riak**, i dijelom **Cassandra**, kako bi omogućili:

* **visoku dostupnost**
* **otpornost na particije**

Omogućuje privremeno pisanje na druge čvorove dok se primarni ne oporavi.



Sloppy quorum omogućuje pisanje na privremeno dostupne čvorove (npr. s3 umjesto s2). Nakon oporavka, podaci se sinkroniziraju (hinted handoff).  
  
  
  
  
  


Podjela ključeva u buckete olakšava usporedbu podataka između poslužitelja. Nedostajući ključevi (npr. "8" na serveru 2) ukazuju na nedosljednost.

Hashiranje ključeva unutar bucketeta stvara jedinstvene oznake. Ažuriranje jednog ključa mijenja hash cijelog bucketa, što olakšava detekciju promjena.

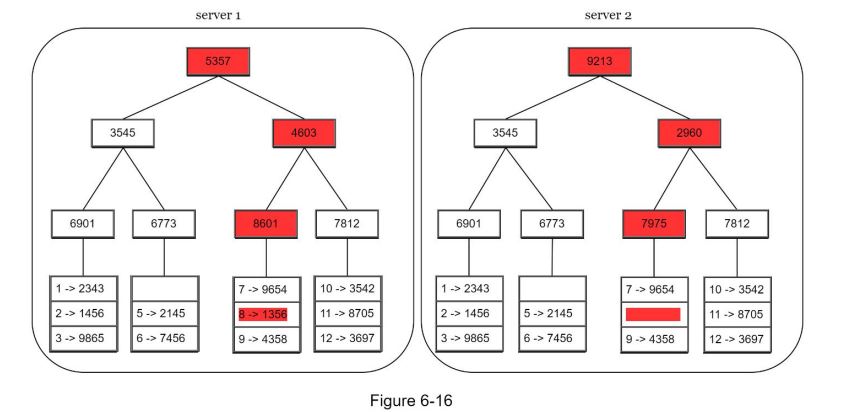
Svaki bucket se sažima u jedan hash čvor (npr. "6901" za bucket 1-3). Usporedba hashova brzo otkriva razlike između poslužitelja.

**Rješavanje nedosljednosti**

**Merkle stabla**

**Merkle stablo je binarno stablo hasheva koje služi za provjeru integriteta podataka.**

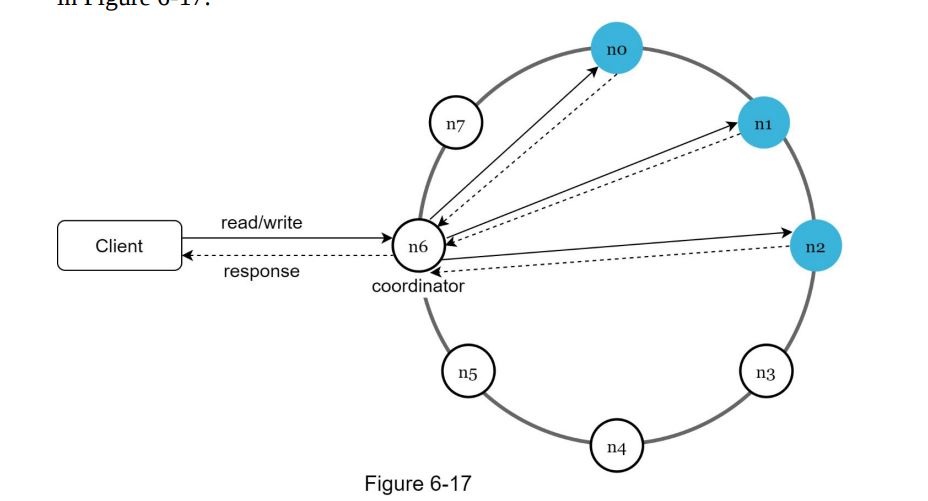
Efikasno uspoređuju podatke između čvorova kako bi se identificirale razlike.



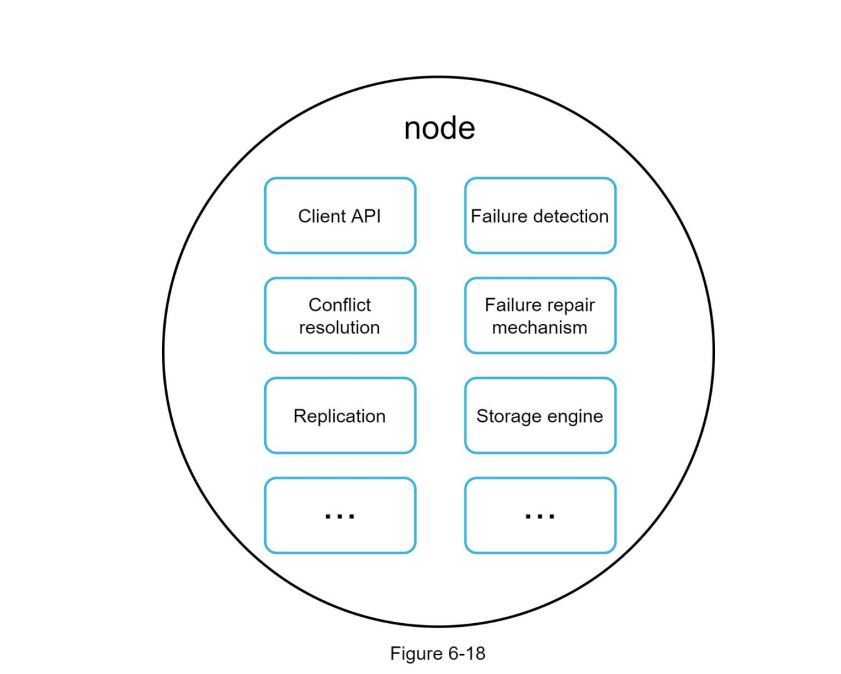
Merkle stablo omogućuje efikasnu detekciju nedosljednosti. Ako se korijenski hash razlikuje, traži se točna grana gdje je došlo do promjene.

**Arhitektura**

Klijenti komuniciraju s koordinatorima, a podaci se pohranjuju na čvorove organizirane u prsten.



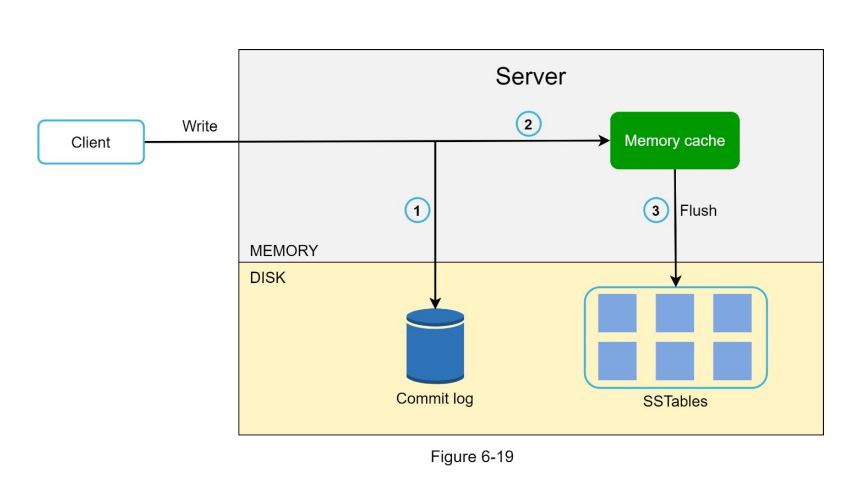
Arhitektura sustava s klijentima, koordinatorima i čvorovima raspoređenim u prsten. Decentralizirani dizajn sprječava usko grlo.



Svaki čvor ima višestruke uloge, uključujući pohranu, replikaciju i detekciju kvarova. Time se izbjegava jedinstvena točka kvara.

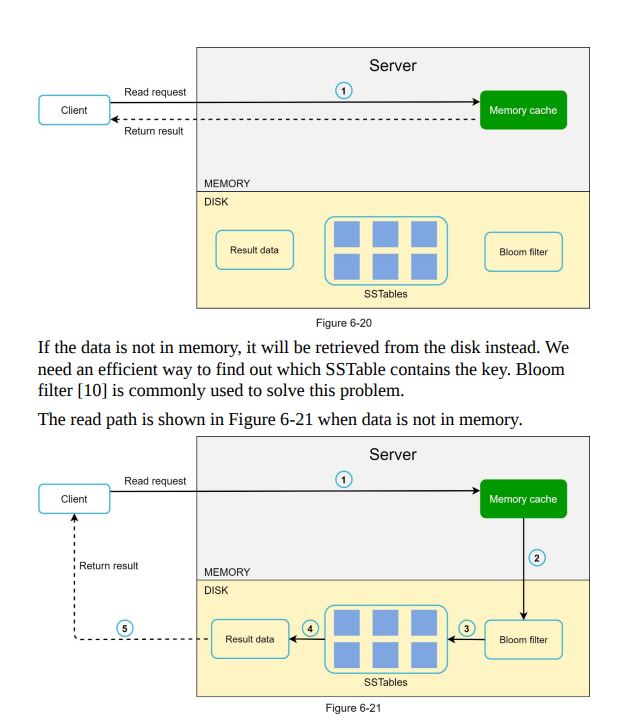
**Pisanje**

Podaci se prvo spremaju u log, zatim u memoriju, pa na disk.



* Put pisanja: podaci se prvo zapisuju u trajni commit log, zatim u memoriju, a na kraju u SSTable na disku. Ovo osigurava otpornost na kvarove.

**Čitanje**   
Ako podaci nisu u memoriji, koristi se Bloom filter za brzo pronalaženje na disku.



1. Put čitanja: ako podaci nisu u memoriji, Bloom filter identificira odgovarajuću SSTable datoteku, smanjujući broj potrebnih I/O operacija.

**Zaključak:** Key-value storeovi poput Dynama ili Cassandra balansiraju između brzine, skalabilnosti i pouzdanosti koristeći gore navedene mehanizme, ovisno o zahtjevima aplikacije (npr. jaka dosljednost vs. visoka dostupnost).