# File Sharing Cloud – dokumentacija

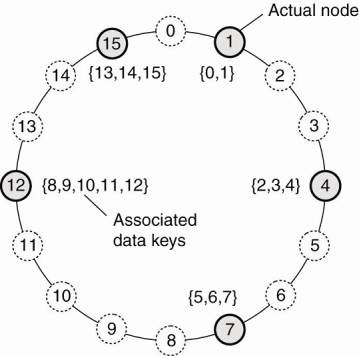
## FSC sistem

File Sharing Cloud predstavlja distribuirani sistem za rad sa ASCII kodiranim tekstualnim datotekama. Datoteke su u njemu organizovane u virtuelnoj strukturi datoteka, gde na svakom pojedinačnom čvoru može da bude prisutan proizvoljan podskup ove čitave strukture. Interakcija sa sistemom se obavlja preko komandne linije. Virtuelni sistem datoteka se čuva dok god postoji barem jedan aktivan čvor. Kada se poslednji čvor ugasi, time prestaje da postoji i virtuelni sistem datoteka (za uredno gašenje čvora se koristi *quit* komanda). Korisniku su omogućene sledeće funkcionalnosti:

* dodavanje nove datoteke sa jedinstvenim nazivom i putanjom (*add <name>* komanda),
* dohvatanje proizvoljne datoteke koja trenutno nije prisutna na čvoru (*pull <name>* komanda),
* uklanjanje datoteke iz sistema (*remove <name>* komanda).

## Arhitektura sistema

Sistem je implementiran uz pomoć distribuirane heš tabele, specifično Chord algoritma. U ovoj implementaciji DHT-a svaki čvor ima svoj ključ dobijen na osnovu heširanja, i zadužen je za određeni opseg vrednosti. Vrednosti nam ovde predstavljaju heširana imena fajlova u virtuelnom sistemu datoteka. Za heširanje se koriste dve proste heš funkcije gde se vrednost čvorovog porta, odnosno ime fajla, množe sa prostim brojevima i potom moduju sa 64.



Svi čvorovi imaju pokazivač na svog prvog sledbenika u lancu, kao i na prvog prethodnika. I upisivanje i dohvatanje ključeva se obavlja tako što se pronađe čvor koji je najbliži sledbenik za taj ključ. Neophodno je da se ova pretraga obavi brzo. Trivijalno bi čvorovi mogli da idu sledbenik-po-sledbenik, ali ovo bi imalo kompleksnost O(N) koja nije prihvatljiva. Rešenje je da svaki čvor drži tablu sledbenika, gde beleži zgodne skokove u lancu. Tipično će se u ovoj tabeli zapisivati vlasnici za ključeve koji su eksponencijalno sve više udaljeni od trenutnog čvora. Često korišćena varijanta bi bila da se u sistemu sa 2n čvorova na svakom čvoru drži tabela sa n stavki, gde na m-tom čvoru za i-tu stavku u tabeli (i uzima vrednosti od 0 do n-1) stoji: *succi = sled((m + 2i) mod 2n)*, gde je *sled* funkcija koja daje najbližeg sledbenika (tj. vlasnika) za traženu heš vrednost.

Kada imamo tabelu sledbenika pravilno iskonstruisanu, koristimo je na sledeći način da pronađemo proizvoljnu DHT vrednost:

* proverimo da li smo mi ili naš sledbenik zaduženi za traženu vrednost;
* ako nismo, pravimo upit ka čvoru u našoj tabeli sledbenika sa najvećim hešom koji nije veći od heša ključa koji se traži;
* proces se ponavlja dok zahtev ne stigne do čvora čiji direktan sledbenik je zadužen za traženu vrednost.

## Bootstrap

Bootstrap server postoji u sistemu i on nije čvor u mreži. Podrazumeva se da je on komponenta sistema koja je uvek uključena. Koristi se isključivo za prvo uključivanje čvora u mrežu. Komunikacija sa njim je veoma ograničena i ona se prekida onog trenutka kada se novom čvoru proslede IP adresa i port nekog čvora iz sistema.

Bootstrap može primiti tri vrste poruka:

* Hail\n<listener\_ip>\n<listener\_port>\n 🡪 Odgovor: <random\_active\_port>\n ili -1
  + Prima je od nekog čvora koji se startuje.
  + Čvoru prosleđuje -1 ako je prvi u sistemu ili port proizvoljnog čvora u sistemu koga novi čvor mora kontaktirati da bi se uključio u sistem.
* New\n<listener\_ip>\n<listener\_port>\n 🡪 Nema odgovora.
  + Prima je od čvora koji je upravo započeo svoju inicijalizaciju.
  + Svrha ove poruke je da novi čvor bootstrap-u da do znanja da nije došlo do kolizije, te ga bootstrap posledično mora uključiti u svoju listu aktivnih čvorova.
* Bye\n<listener\_ip>\n<listener\_port>\n 🡪 Nema odgovora.
  + Prima je od čvora koji napušta sistem.
  + Nakon primanja ove poruke, napuštajući čvor se uklanja iz liste aktivnih čvorova.

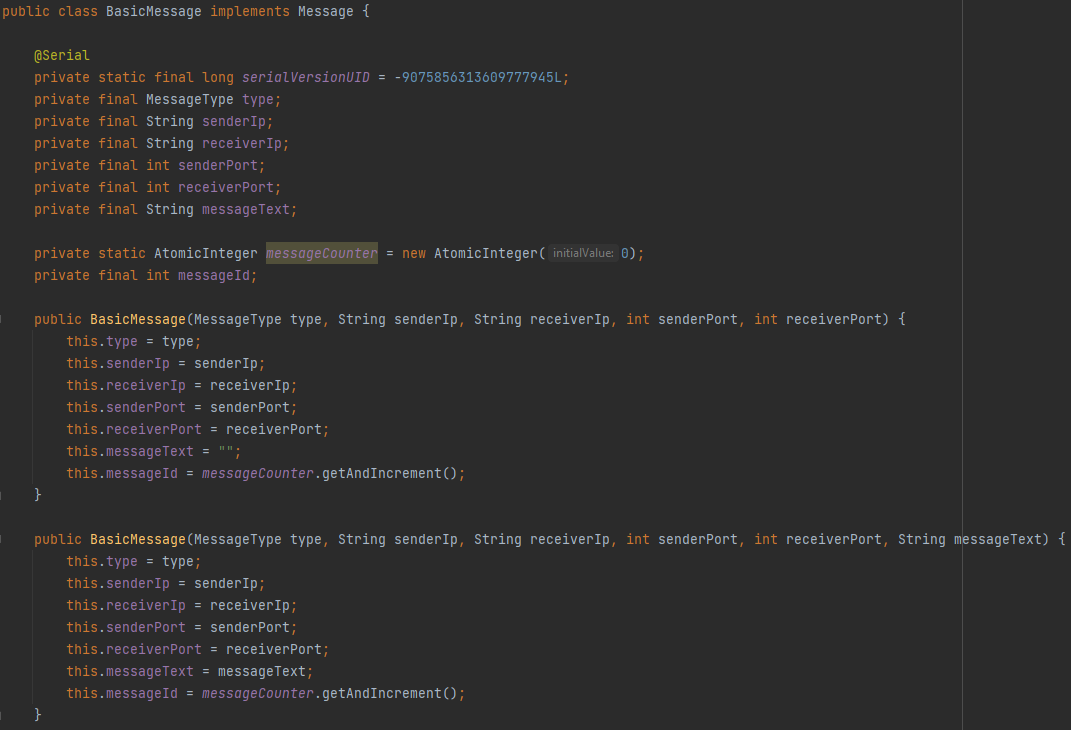
## Poruke u sistemu

### AddMessage



Add poruka se šalje pri izvršavanju *add <name>* komande. Uslovi da bi ona bila poslata su sledeći. Datoteka koja se dodaje u virtuelni sistem datoteka mora biti prisutna na čvoru na kom se izvršava komanda, a ujedno taj čvor ne sme biti vlasnik ključa za tu datoteku (jer ako jeste, onda će sam dodati datoteku u sistem). Ova poruka se prosleđuje dok ne stigne do čvora odgovornog za ključ datoteke koja se dodaje. Taj čvor onda vrši dodavanje datoteke u sistem.

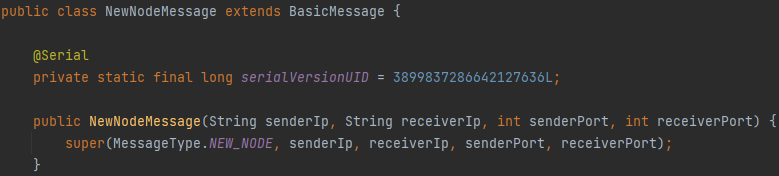
### BasicMessage



BasicMessage poruka predstavlja osnovni format za sve poruke u sistemu. Kao što se može videti na slici, svaka poruka sadrži barem sledeće informacije:

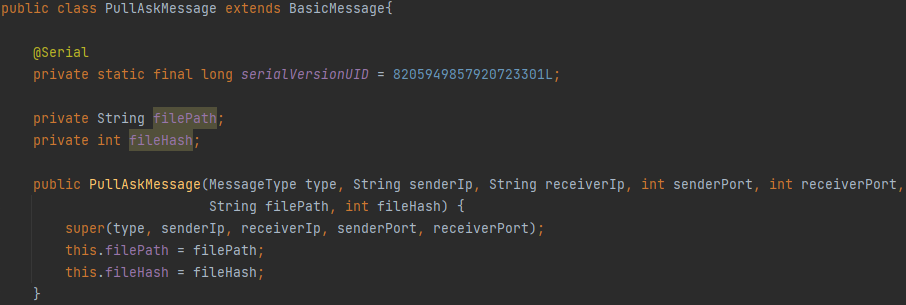
* tip poruke,
* IP pošiljaoca,
* IP primaoca,
* port pošiljaoca,
* port primaoca,
* tekst poruke,
* ID poruke.

### NewNodeMessage



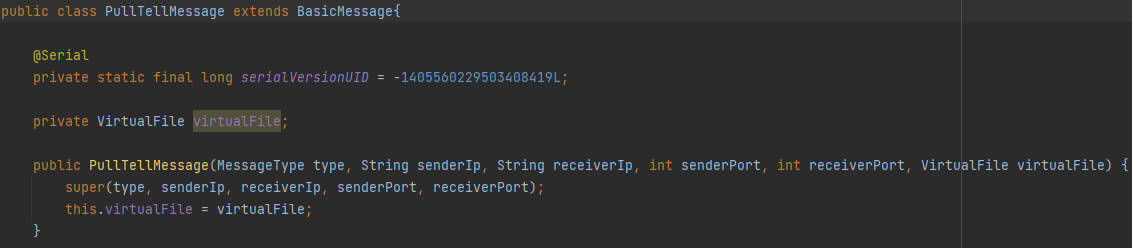
NewNode poruka se šalje pri startovanju čvora koji nije prvi čvor u sistemu. On se obraća bootstrap čvoru i od njega dobija informacije o čvoru kome se mora obratiti da bi ga isti startovao. Tom čvoru onda šalje ovu poruku. Kada poruka stigne do primaoca, proverava se da li bi uključivanjem novog čvora u sistem moglo doći do kolizije, kao i da li je pošiljalac njegov prethodnik. Ako nema kolizije i on jeste njegov prethodnik, onda ga on uključuje u sistem, a ako nije, prosleđuje poruku napred (ona mora stići do čvora kome novi čvor jeste prethodnik, jer će ga isti uključiti u sistem).

### PullAskMessage



PullAsk poruka se šalje pri izvršavanju *pull <name>* komande. Ona se prosleđuje do čvora zaduženog za datoteku koja se dohvata iz virtuelnog sistema datoteka. Nakon pristizanja do tog čvora, datoteka se dohvata (u slučaju da se zapravo nalazi u sistemu) i šalje nazad čvoru koji je tražio (unutar PullTell poruke). U slučaju da je čvor na kome se izvršava pull komanda takođe i čvor koji je zadužen za ključ tražene datoteke, onda korisnik biva obavešten da mora datoteku dodati u sistem, umesto dohvatiti.

### PullTellMessage



PullTell poruka se šalje kao odgovor na prethodno primljenu PullAsk poruku. Ona je uvek poslata ka pošiljaocu PullAsk poruke i sadrži traženu virtuelnu datoteku, u slučaju da je ista pronađena u virtuelnom sistemu datoteka.

### QuitMessage



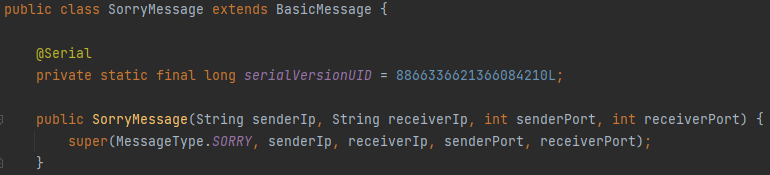
Quit poruka se šalje pri izvršavanju *quit* komande. Slično kao i kod Update poruke, i ovde je svrha poruke da prođe kroz ceo sistem i da svim čvorovima do znanja da se jedan od njih gasi. Njegov sledbenik će biti taj koji preuzima sve njegove vrednosti. On, kao i svi naredni čvorovi na putanji, će takođe osvežiti svoju tabelu sledbenika. Kada Quit poruka napravi krug i stigne nazad na čvor koji napušta sistem, on će isprazniti svoj radni direktorijum i obavestiti bootstrap čvor da ga mora ukloniti iz liste aktivnih čvorova. Nakon toga, on se gasi.

### RemoveMessage



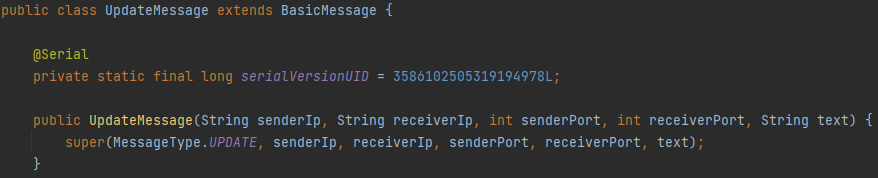
Remove poruka se šalje pri izvršavanju *remove <name>* komande. Ako čvor na kome se izvršava komanda nije zadužen za ključ datoteke koja se briše, onda se poruka prosleđuje napred. Nakon pristizanja na čvor koji jeste zadužen za tu datoteku, vrši se provera da li se ona uopšte nalazi u virtuelnom sistemu datoteka. Ako se nalazi, onda se briše iz sistema.

### SorryMessage



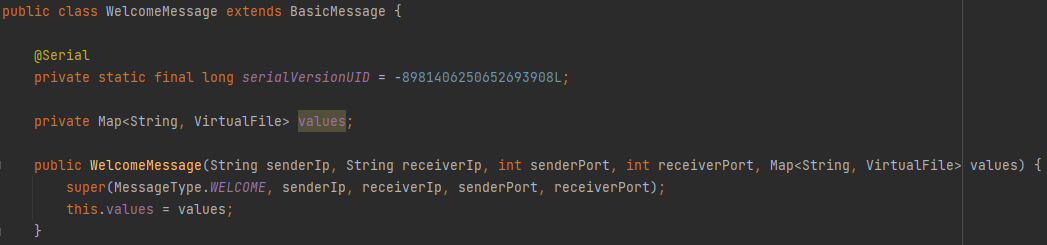
Ako je pri uključivanju novog čvora u sistem došlo do kolizije (njegov ID se poklapa sa ID-em jednog od postojećih čvorova), onda će čvor koji ga uključuje novom čvoru poslati Sorry poruku. Nakon primanja te poruke, novi čvor će odustati od svog pokušaja startovanja i ugasiti se.

### UpdateMessage



Update poruka se šalje nakon uspešno završenog uključivanja novog čvora u sistem. Ona prolazi kroz ceo sistem. Svoju putanju počinje i završava na novom čvoru, a u međuvremenu posećuje i sve ostale čvorove. Ovako svi čvorovi u sistemu postaju svesni dolaska novog čvora, dok ujedno i on postaje svestan svih drugih čvorova (dodaju se odgovarajuće vrednosti u tabeli sledbenika).

### WelcomeMessage



Welcome poruka se šalje novom čvoru nakon toga što je čvor kome je on prethodnik proverio da nema kolizije, i odredio koje od svojih vrednosti mu mora predati. Kada novi čvor dobije ovu poruku, on započne svoju inicijalizaciju.