Dokumentacija

Ilija Dimitrijević RN9920

1. **Serventi**

* Host
* Port
* Id
* Key

1. **Poruka**

Svaka komunikacija između servenata u ovom sistemu odvija se putem poruka. Poruka je apstraktan entitet i svaka specifična vrsta poruke nasleđuje ovu klasu. Poruka je serijalizovana kako bi se slala preko mreže.

Glavne karakteristike poruke su:

* Id: Jedinstveni identifikator poruke, generisan koristeći AtomicInteger. Svaka nova poruka inkrementira ovaj broj kako bi dobila svoj jedinstveni ID.
* Text: Opcionalno polje koje sadrži proizvoljan tekst koji se šalje kroz poruku.
* Sender: Servent koji šalje poruku.
* Receiver: Servent kome je poruka namenjena.
* Route: Lista servenata koji su poslali ovu poruku, pružajući trasu kako je poruka putovala od početka do krajnjeg serventa.

Svaka specifična poruka (koja nasledjuje Message), implementira handle() metodu, koja određuje šta se dešava kada poruka stigne do svog odredišta.

**Tip poruka:**

* AddMessage
* CheckedMessage
* CheckMessage
* FailMessage
* JoinAskMessage
* JoinTellMessage
* PingMessage
* PongMessage
* PullAskMessage
* PullTellMessage
* RemoveMessage
* RemovedMessage

Sistem takodje sadrži MessageHandler I MessageListenner klase koje rade u zasebnim Threadov-ima. Kada poruka stigne, MessageListenner je procita iz socket-a i dodaje je u BlockingQueue, a MessageHandler izvlaci poruke iz tog reda, i prosledjuje ih odgovarajucem handleru, koji je odgovoran za taj tip poruke.

1. **Komunikacija Servenata i priključivanje u mrežu**

Za komunikaciju servenata koristi se specifičan sistem implementiran pomoću Kademlia algoritma.

Kada novi servent (npr. Servent1) želi da se poveže u sistem, on se prvo prijavljuje bootstrap čvoru koji je postavljen na localhost:10000. Bootstrap čvor je odgovoran za inicijalno uspostavljanje veze. Nakon toga, bootstrap čvor dodaje novi servent u skup svih trenutno dostupnih servenata i generiše ključ novog serventa.

Da bi novi servent dobio svoj ključ, koristi se SHA-1 algoritam. Uzimamo serventov heš kod, pretvaramo ga u niz bajtova i onda ga "provlačimo" kroz SHA-1 algoritam. Na kraju, dobijamo BigInteger, koji će biti jedinstveni ključ za taj servent.

Zatim, bootstrap čvor traži najbliže servente novom serventu koristeći XOR udaljenost. Kada novi servent dobije listu najbližih servenata, dodaje ih u svoj set servenata i šalje im "Ping" poruke. U koliko nemaju servent od kog su primili Ping poruku, dodaju ga u Set<Servents> servents, kako bi se povezali sa njim.

Ako servent dobije "Ping" poruku, on zauzvrat šalje "Pong" poruku da potvrdi svoju prisutnost u mreži. Kako bi se obezbedila stalna dostupnost servenata, implementiran je "Servent Pinger" thread. On kontinualno šalje "Ping" poruke svim serventima u setu (svim serventima sa kojima on komunicira) i čeka "Pong" poruke od njih.

Ako neki servent ne vrati "Pong" poruku u predviđenom vremenskom okviru (FAILURE\_SOFT interval), proverava se njegova dostupnost tako što se od najbližeg serventa traži da ga proveri. Ako servent ne odgovori ni nakon ove provere, čeka se da FAILURE\_HARD interval. Ako nakon tog intervala servent ne odgovri sa “Pong “porukom, smatra se da je servent ispao iz sistema.

Servent koji je zaključio njegovo ispadanje, uklanja servent iz svog seta I javlja bootstrap čvoru kako bi ga uklonio iz liste svih servenata u sistemu.