DOKUMENTACIJA PROGRAMA LOADBALANCER

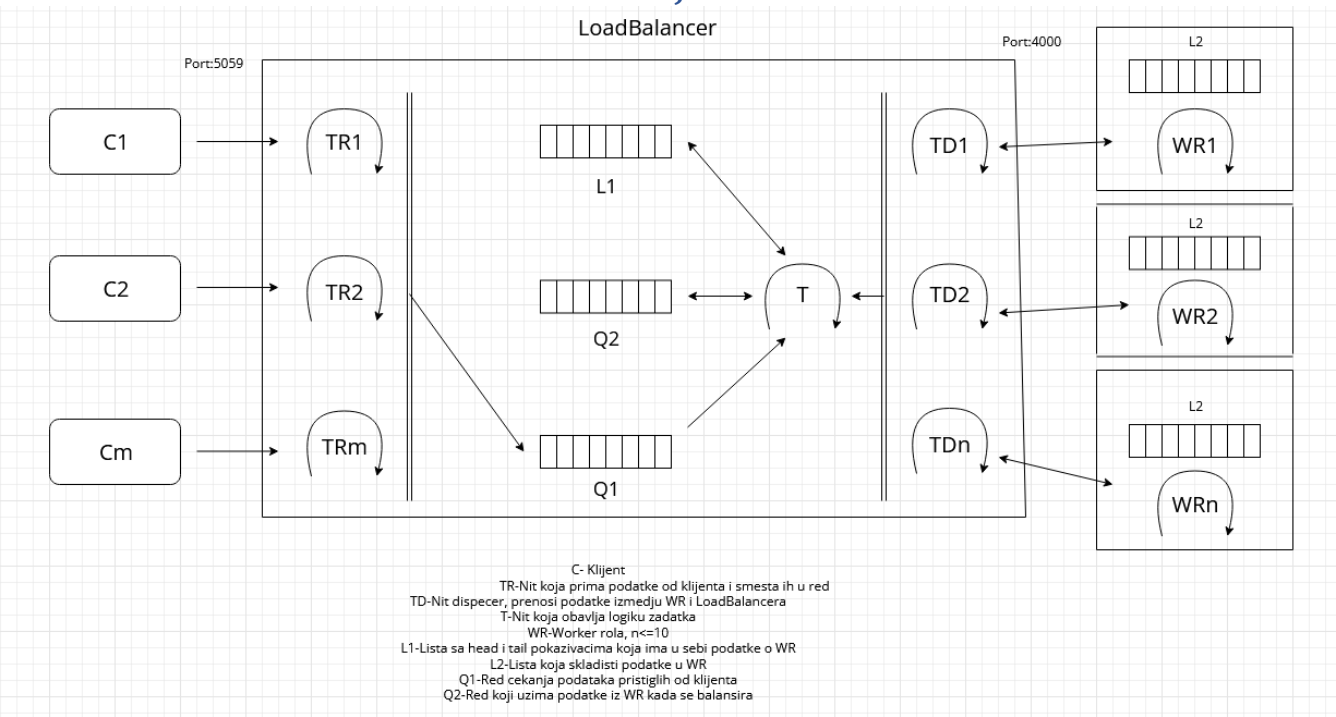
DUŠAN KNEŽEVIĆ PR74/2019

UROŠ TEŠIĆ PR89/2019

1. UVOD

Razviti servis koji vrši skladištenje podataka dobijenih od klijenata. Servis se sastoji od dva tipa procesa - Load Balancer (LB) i Worker (WR). Postoji jedna instanca LB komponenti koja sluša na portu 5059 i prima zahteve za skladištenje. Svaki zahtev prosleđuje onom WR koji ima najmanje skladištenih podataka. WR nakon skladištenja podataka javlja LB da je završio. Prilikom prijavljivanja novog WR, podaci se distribuiraju tako da svi klijenti imaju istu količinu podataka. Potrebno je izmeriti propusnost sistema sa 1-10 WR. LB dobija informaciju o broju WR kojima raspolaže tako što se oni prilikom startovanja registruju LB odnosno prilikom gašenja odjave. LB takođe ima podatak koliko svaki WR ima skladištenih podataka.

2. DIZAJN



3. REŠENJE

Aplikacije funkcioniše tako što se klijenti prijavljuju na servis preko porta 5059 i šalju podatke fiksne dužine. Worker Role se prijavljuju na LoadBalancer preko porta 4000 i tom prilikom ostavljaju svoje podatke (kritične sekcije, accepted socket, broj primljenih podataka) koji se čuvaju u listi L1.
Pristigle podatke od strane klijenata LoadBalancer smešta u red “Q”. Nit “T” proverava da li u privremenoj memoriji (Lista L2) ima nekih podataka, u slučaju da nema izvršava logiku distribuiranja na osnovu podataka iz liste L2, uzima podatke iz reda i preko niti dispečer “TD” šalje Worker Roli koja je najmanje opterećena podacima. Prilikom odgovora Worker Role podaci o stanju skladišta u listi L1 se ažuriraju.
Kada se prijavi nova Worker Rola vrši se balansiranje skladišta svih Worker Rola tako što na osnovu logike sistema sve Worker Role pošalju odredjeni broj podataka koji se smešta u privremenoj memoriji, Lista L2, i zatim se ti podaci pošalju novoj Worker Roli. Isti princip se izvodi ukoliko se neka Worker Rola odjavila sa LoadBalancera.

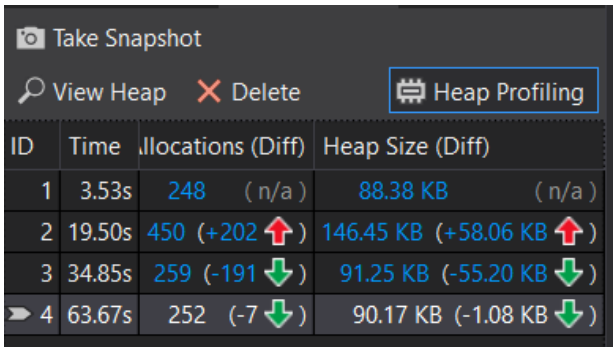
4. STRUKTURE PODATAKA

Redovi (Q1, Q2) se koriste za skladištenje privremenih podataka jer je to struktura koja omogućuje lako i brzo ubacivanje i izbacivanje podataka, kao i zato što predstavlja red čekanja za podatke.

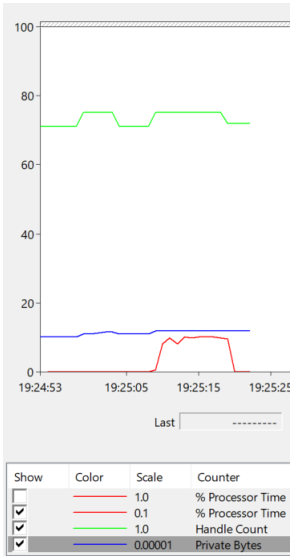
Lista sa head i tail pokazivačima se koristi na mjestima gdje nije bitna brzina smještanja podataka već sama organizacija njih. Listu koristimo kako bi lakše podatke prebacivali sa kraja na početak i obrnuto što nam je ključno za distribuciju podataka.

5. TESTIRANJE

U okviru testiranja projekta je odradjen Stress Test koji je implementiran tako da klijent u sekundi šalje 200 poruka i opterećuje redove za skladištenje podataka. Na priloženim slikama je prikazano stanje heap-a prilikom testiranja projekta kao i korišćenje procerskog vremena, handle-ova i memorije računara.



Snapshot 1. – Pokrenut program.
Snapshot 2. – Početak stress testa, poslato 200 poruka sa klijenta.
Snapshot 3. – Pokrenut worker, odradjena distribucija.
Snapshot 4. – Kraj programa.



6. ZAKLJUČAK

Rezultati testova su skoro očekivani. Na početku programa je zauzeto više memorije nego što je očekivano, sam rad programa ima dobru kontrolu memorije, pravovremeno oslobadja zauzete resurse. Na kraju programa ostane zauzeto više memorije nego što je to bio slučaj na početku programa, a to je zbog handle-ova koji kada se zatvore ne briše se njihova memorija.