**Load Balancer** (LB-G1)

UVOD

Uvođenjem pametnih brojila nastaje problem dugog vreme proračuna potrosnje računa. Cilj ovog projekta je simulacija rada Load Balancer sistema za obracun mesecne potrosnje struje pametnih metera, kako bi raspodelili opterecenje proracuna i ubrzali proces za vece kolicine klijenata. Ovaj sistem ima primetne benefite kod većeg broja brojila, dok je za menje brojila to manje primetno.

DIZAJN

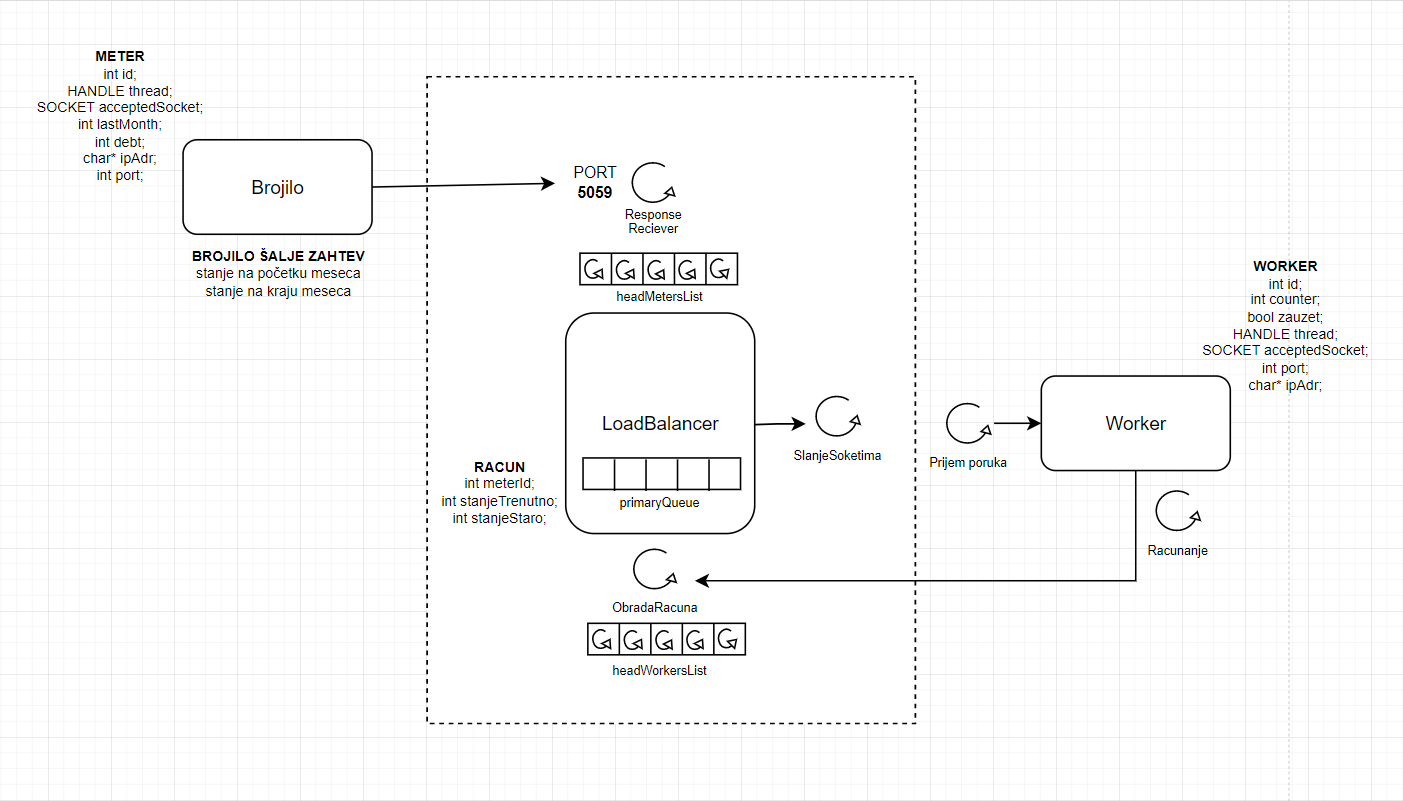
Aplikacija se sastoji od 3 tipa komponente: *LoadBalancer, Worker, Meter*.

Prilikom prijave na sistem meter se javlja balanceru na portu 5059 sa ciljem da se registruje gde ga on prihvata i cuva u headMeterList. Ostvaruje se konekcija. Svakom meteru dodeljuje novi thread koji osluskuje na soketu racune koje ce registrovani meteri slati i zatim ih čuva u primaryQueue tipa RingBuffer-a.

Sa druge strane jedna nit očekuje poruke i zahteve za registraciju novih workera koji ce biti na raspolaganju Load Balanceru u headWorkersList redu.

Ukoliko se u primaryQueue pojave poruke i imamo workera na raspolaganju naša nit (funkcija Slanje soketima) uzima prvi slobodan worker i sledeci Račun koji je na redu i racuna ga i ceka odgovor u niti ObradaRacuna workera koja radi samo dok ne dobije odgovor. Promeni zaduzenje u strukturi brojila i nit se zatvara.

Worker samo čeka zahteve i vrsi proračune. Dug u toku meseca koji je dobio računanjem salje nazad LB kako bi ga on sačuvao. (zaduzenje se računa tako sto trenutnoStanje-stanjeProslogMeseca pomnozimo sa cenom 1KWh, npr. u našem slučaju 117 dinara).



STRUKTURE PODATAKA

U projektu su korišćene strukture podataka *RingBuffer, List/Queue.*

RingBuffer (Circular Buffer) je FIFO struktura koja skladišti pristigle zahteve za računanje tj. račune sa strane brojila, čuva ih i kada se stvori prilika, šalje ih ka slobodnom workeru na obradu. Koristimo ga jer želimo da šaljemo zahteve redom, a ukoliko dođe do zakrčena i punjenja buffera mozemo se vratiti na pocetak i opet obračunati dug, a nece doci do gubitka racuna jer racun cuva trenutnu vrednosti i vrednost na kraju proslog obracunskog perioda iz metera.

Lista je takođe FIFO struktura u koju skladištimo Metere i Workere i olakšava nam pretragu i prebrojavanje istih.

OPISIVANJE NAJBITNIJIH FUNKCIJA

*Funkcije na workeru:*

void SetNonblocking(SOCKET\* socket) – funkcija koja se obavlja na postojecim soketima i ona soket konfigurise u neblokirajući režim. Takođe postoji funkcija koja soket prebacuje u blokirajući režim.

SOCKET SetConnectedSocket(u\_short port) – funkija koja inicijalizuje soket na zadatom portu i vraća ga.

*Funkcije na meteru:*

SOCKET SetConnectedSocket(u\_short port) – funkcija koja inicijalizuje soket na zadatom portu i vraca vrednost soketa.

int GenerateRandomNumber(int minNumber) – funkcija koja random generiše int vrednost, koja je veca od zadate vrednosti.

*Funkcije na LB:*

DWORD WINAPI WorkWithSockets(void\* vargp) – funkcija koja prati dešavanja na prosleđenom soketu, selektujemo ga i pratimo odgovore. Ukoliko pristigne poruka zahteva za konekciju povratna vrednost selekcije bude > 0 i znamo da je pristigao neki zahtev. Pravimo novi meter i cuvamo ga na kraju liste.

DWORD WINAPI PrijemDaljihPoruka(void\* vargp) – funkcija koju pokreće WorkWithSockets u zasebnim tredovima koje čuvamo u meteru. Svrha funkcije je da očekuje poruke od registrovanih metera i da pravi račune koje ce čuvati u RingBuffer.

DWORD WINAPI WorkWithSocketsWorker(void\* vargp) – funkcija koja osluškuje na drugom soketu i čeka zahteve za registraciju novih workera koje kreira i čuva u novi red.

void SlanjeSoketima() – funkcija koju pozivamo u main niti, njena uloga je da proverava broj workera i broj računa koji cekaju na računanje. Ukoliko ih imamo vršimo preuzimanje računa iz buffera i slanje prvom slobodnom workeru. Worker u svojoj strukturi dobija thread koji poziva funkciju ObradaRačuna(worker).

DWORD WINAPI ObradaRacuna(void\* vargp) - pozivamo je za slobodnog workera koje ga zauzme tako što property zauzet postavi na true i posalje mu zadatak u formatu METER ID/STANJE STARO/STANJE NOVO. Zatim primi odgovor od workera i pozove funkciju UvecajDug koja izmeni meter i oslobodi workera. Tu se ta nit, završava do slede upotrebe tog workera.

ZAKLJUČAK

Rešenje ovog zadatka je rešeno jednostavnom komunikacijom pomocu TCP transportnog protokola. Prilagođeno je istovremenom radu sa više workera i metera zahvaljujući upotrebi threadova. LoadBalancer pamti listu svih workera i metera kako bi vodio evidenckiju o njima i znao kome da šalje zaduženja. Ovo je ostvareno uz pomoc strukturiranja unutar lista i kružnog buffera.

POTENCIJALNA UNAPREĐENJA

Potencijalno rad sa redom u kojem čuvamo metere mogli bi zameniti sa čuvanjem u neki fajl ili bazu podataka. Takođe u bi se funkcije metera mogle unaprediti sa plaćanjem i umanjenjem zaduženja kod Load Balancera za taj meter.

Stefan Popovic PR75/2019

Rastisla Zaskalicki PR77/2019