Fakultet tehničkih nauka

Univerzitet u Novom Sadu

Load Balancing

Projekat 13 - Industrijski komunikacioni protokoli

PR103/2020 Anabela Zonai
PR95/2020 Kristina Sretenović

13. januar 2024.

UVOD

Opis problema koji se rešava i ciljevi zadatka

Opis problema obuhvata razvoj servisa za dinamičko balansiranje opterećenja između Load Balancer (LB) i Worker (WR) komponenti. Load Balancer sluša na portu 5059 i zadužen je za primanje zahteva za obradu. Takođe, upravlja zahtevima koji ulaze u sistem i vrši njihovo prosleđivanje radnicima (Workerima).
Kako bismo pojednostavili ovaj primer možemo ga poistovetiti sa situacijom u kom imamo više radnika koji upravljaju poslovima koji sa nalaze u nekom redu.

Kontrola broja radnika: Ako ima malo poslova (manje od 30% popunjenosti reda), ne dodaje nove radnike. Samo postojeći radnici rade.

Povećanje broja radnika: Kad ima puno poslova (preko 70% popunjenosti reda), dodaje nove radnike da pomognu u obradi. Ovo se dešava kad je gužva.

Zaustavljanje prihvatanja novih poslova: Ako je red potpuno pun, neće prihvatiti nove poslove sve dok se neki ne završe i ne oslobodi se mesto.

Smanjenje broja radnika: Ako ima malo poslova (ispod 30% popunjenosti reda), postepeno smanjuje broj radnika, tako da na kraju ostane samo jedan radnik.

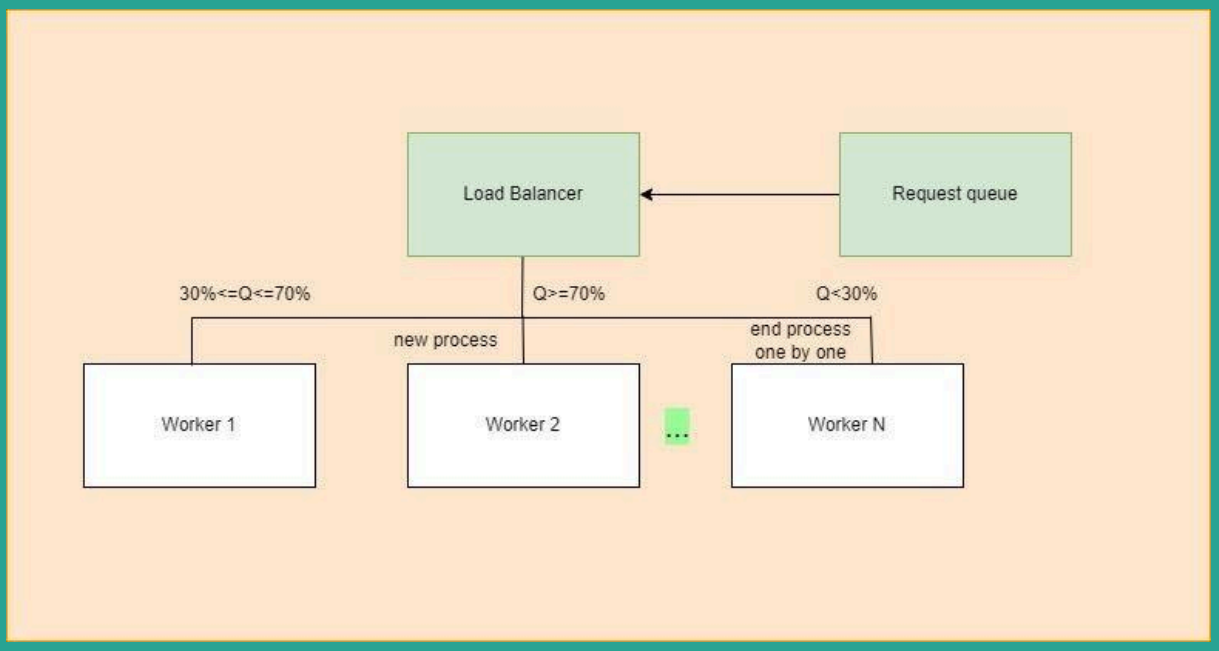
Kada je popunjenost reda između 30% i 70%, servis održava stabilnost u broju radnika. To znači da ne dodaje nove radnike i ne gasi postojeće, već radi sa postojećim brojem radnika.
Servis jednostavno održava stabilnost u broju radnika kako bi obradio poslove na normalan i efikasan način, bez preteranog proširivanja ili smanjivanja radne snage.

Ciljevi sistema

- Efikasno upravljanje zahtevima koji ulaze u sistem.
- Dinamičko prilagođavanje broja radnih instanci (WR) u skladu sa opterećenjem.
- Obezbeđivanje da sistem može da se širi ili smanjuje u skladu sa promenama u opterećenju.
- Ovaj sistem ima za cilj da efikasno upravlja obradom zahteva i skalira resurse (WR instance) u skladu sa opterećenjem sistema, omogućavajući prilagođavanje broja aktivnih WR instanci u zavisnosti od trenutnog opterećenja i kapaciteta sistema.

DIZAJN

Opis dizajna implementiranog rešenja, dijagram komponenti



Servis se sastoji od dva tipa procesa - Load Balancer (LB) i Worker (WR). Load Balancer prima zahteve za obradu iz reda zahteva i ima jednu instancu. Worker može imati više instanci u zavisnosti od popunjenosti reda Q. Ukoliko je popunjenost Q između 30% i 70% od predefinisane veličine broj WR instanci se ne menja. Ako Q krene da raste preko 70% - pravi se nova instanca Worker-a. Ukoliko se popunjenost reda Q spusti ispod 30%, treba gasiti postepeno WR instance, sve dok ne ostane samo jedna instanca.

Strukture podataka

Razlozi zbog kojih su izabrane baš te strukture podataka

U implementaciji ovog zadatka koristile smo strukturu reda (queue). Korišćenje strukture reda za ovaj tip problema dinamičkog balansiranja opterećenja ima nekoliko prednosti:

- FIFO pristup (First In First Out)
    - Struktura reda koristi ovaj pristup što znači da zahtevi koji prvi stignu u red čekanja će biti prvi obrađeni.
    - FIFO pristup je koristan u situacijama gde je redosled obrade bitan, što često i jeste slučaj kod zahteva koji pristižu u sistem.
- Jednostavno upravljanje redom čekanja
    - Queue struktura omogućava jednostavno dodavanje novih zahteva na kraj reda i uklanjanje zahteva sa početka reda, što olakšava upravljanje zahtevima sistema.
- Poboljšana kontrola opterećenja
    - Red čekanja omogućava da se popunjenost (broj zahteva) jednostavno prati. Time je lako kontrolisati trenutno opterećenje sistema.
- Lakše praćenje sistema
    - Kroz red čekanja je lakše pratiti koliko zahteva trenutno čeka na obradu. Na taj način se olakšava odluka o dodavanju ili uklanjanju WR instanci.

Opis i pojašnjenje semantike podataka koje sadrže strukture

Podaci u strukturi reda imaju određenu semantiku ili značenje koje se koristi za upravljanje zahtevima i radom sistema. Evo par opisa i pojašnjenja semantike podataka unutar strukture reda:

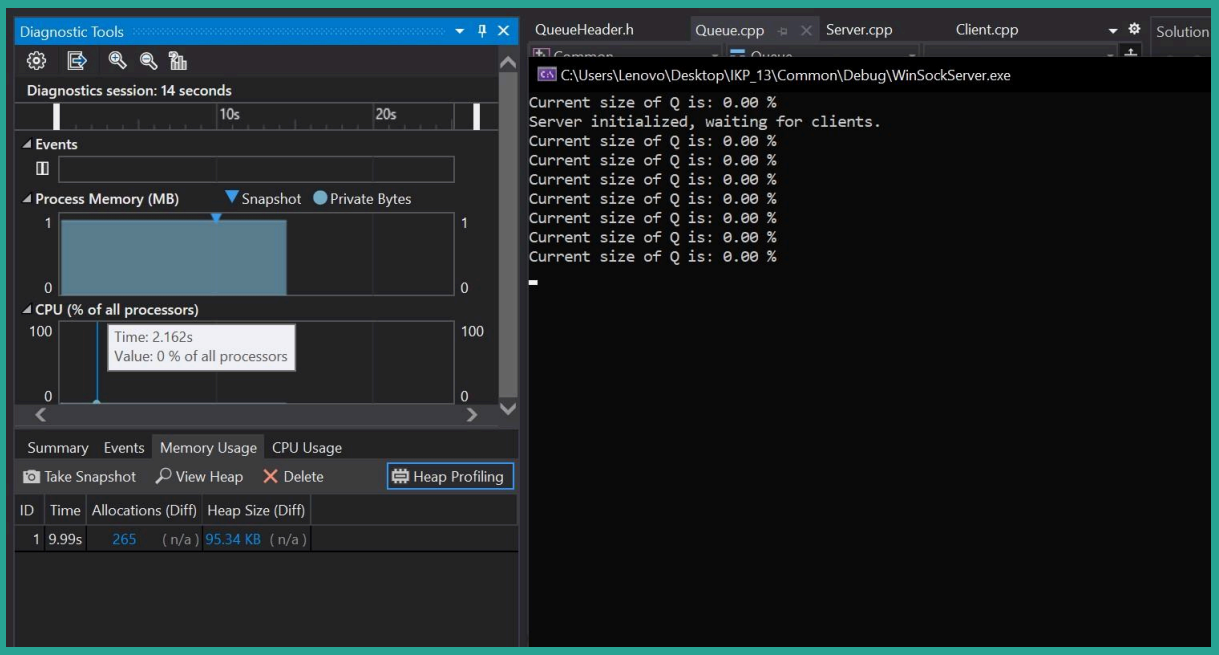
- Zahtevi za obradu
    - Podaci o zahtevima: Svaki zahtev koji se dodaje u red čekanja sadrži informacije koje su potrebne za obradu (parametri zadatka, prioritet…).
    - Struktura podataka za zahtev: Moguće je da svaki zahtev bude predstavljen strukturom podataka koja sadrži informacije o zadatku koji treba obaviti.
- Red čekanja
    - Semantika reda: FIFO semantika što znači da se zahtevi obrađuju prema redosledu dodavanja u red.
    - Kontrola popunjenosti: Popunjenost reda se prati kako bi se procenilo opterećenje sistema. Ako je Q između 30% i 70%, ne menja se broj WR instanci.
- Dinamičko dodavanje i uklanjanje zahteva:
    - Dodavanje novih zahteva: Novi zahtevi se dodaju na kraj reda.

    - Uklanjanje zahtevaČ Kada se zahtev obradi odnosno kada WR instanca završi obradu, zahtev se uklanja sa početka reda.
- Odluke o skaliranju
    - Praćenje popunjenosti: Popunjenost reda čekanja Q je indikator trenutnog opterećenja sistema.
    - Odluke o skaliranju: Na osnovu popunjenosti reda Q, odluke se donose o pokretanju ili gašenju WR instanci radi prilagođavanja opterećenju sistema.

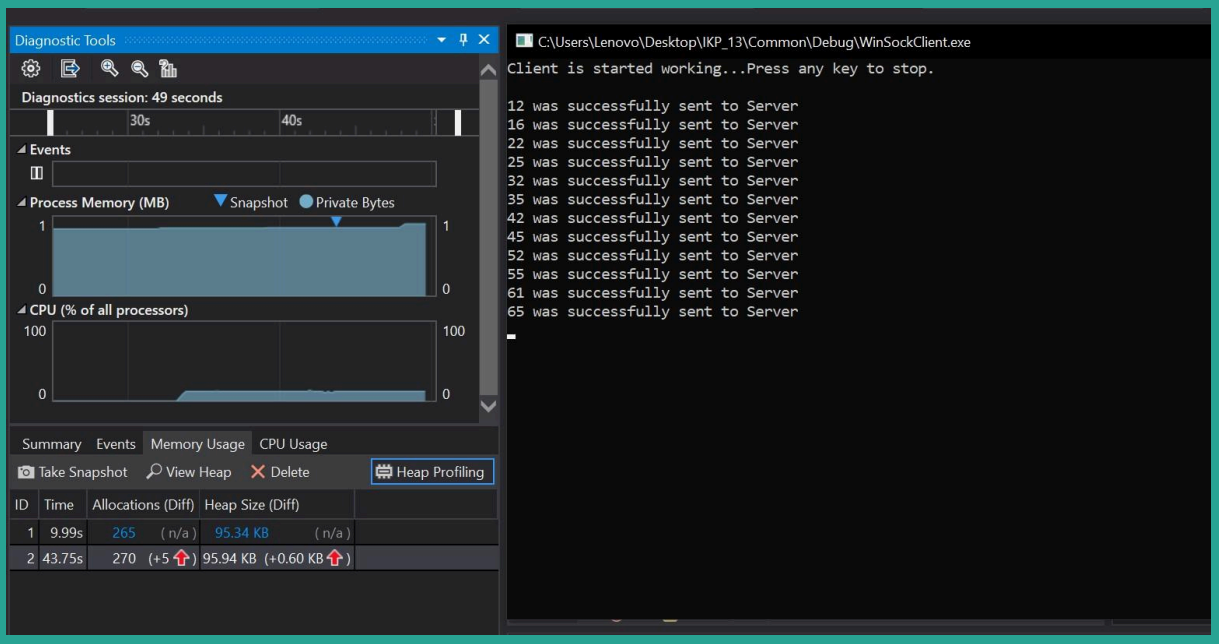
Rezultati testiranja

U okviru ovog zadatka izvršeno je testiranje.

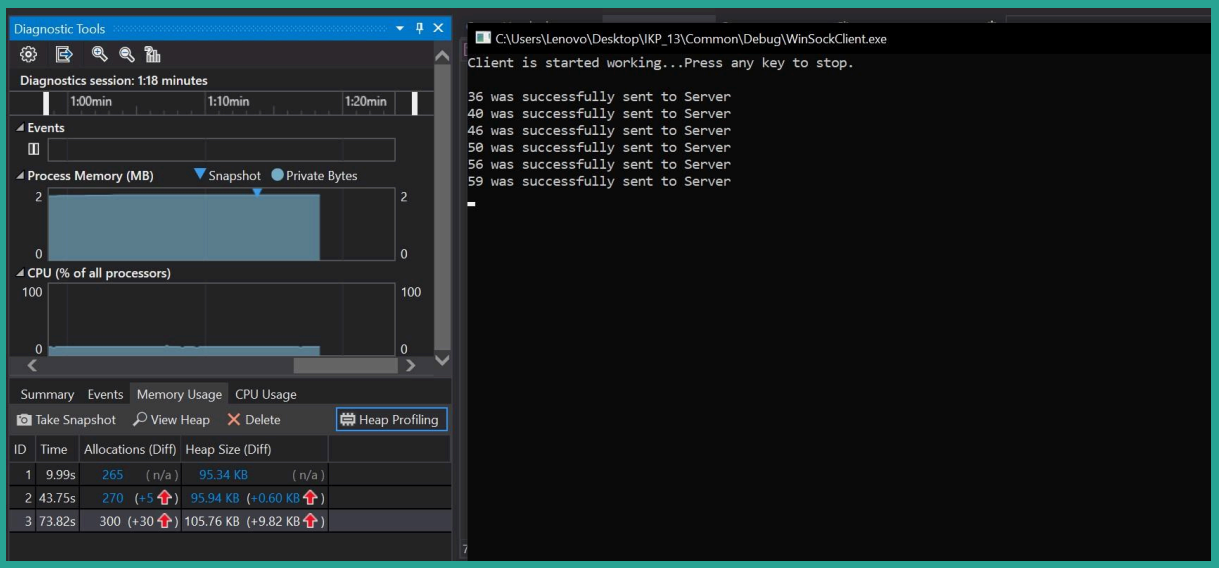
- Prvo testiranje je snapshot gde je pokrenut samo server:



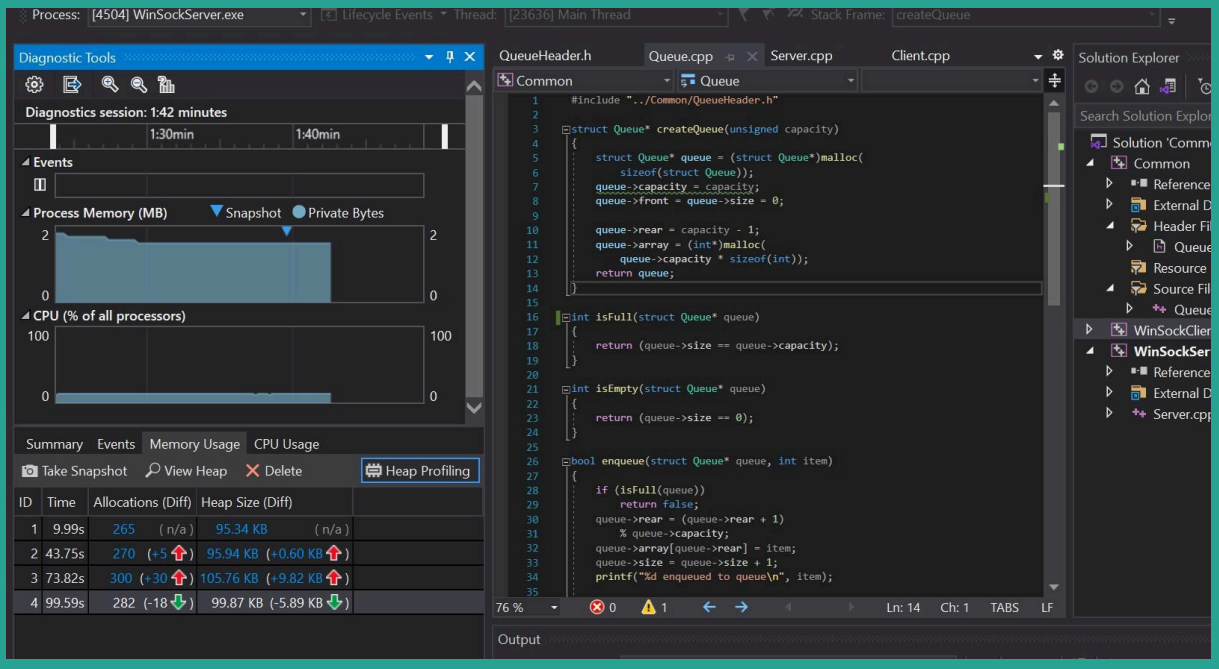
- Drugo testiranje je snapshot gde je pokrenut 1 klijent:



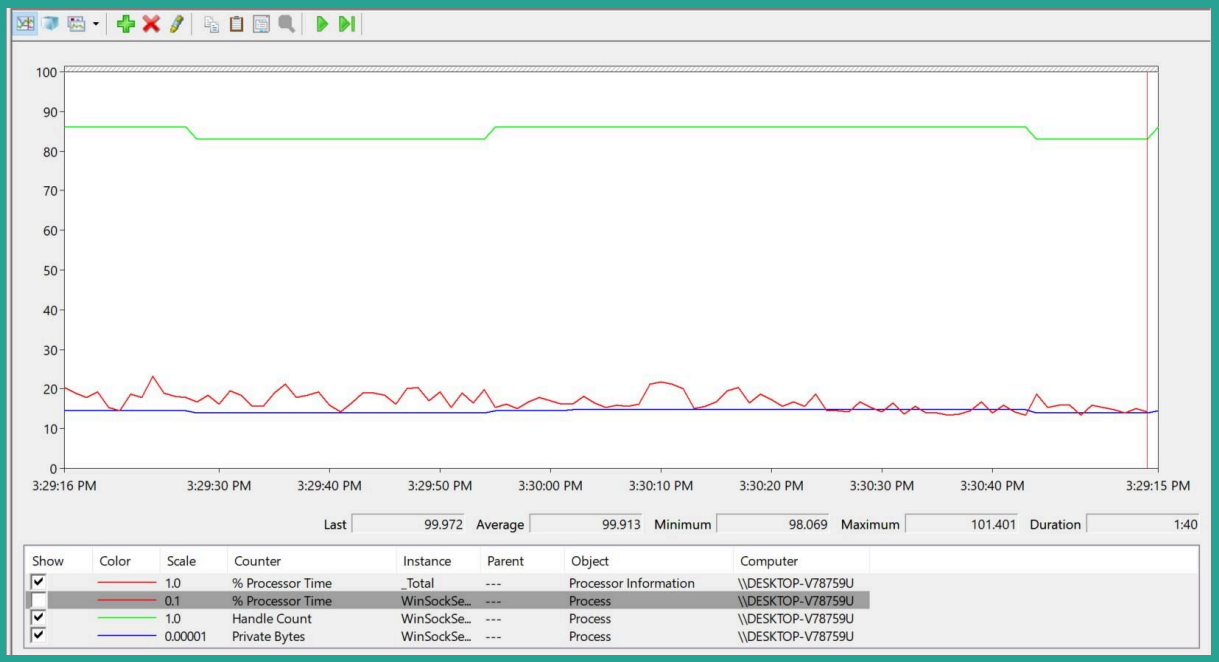
- Treće testiranje je snapshot gde je pokrenuto 3 klijenta:



- Četvrto testiranje je snapshot gde su ugašena 2 klijenta, 1 ostao upaljen:



- Peto testiranje je urađeno preko Performance Monitor-a. Istestirane su opcije Handle Count, Processor Time i Private Bytes:



Zaključak

Na osnovu rezultata testiranja, možemo izvući nekoliko zaključaka:

- Efikasnost sistema: Preliminarna testiranja pokazuju zadovoljavajuću efikasnost sistema u upravljanju zahtevima i dinamičkom prilagođavanju broja Worker instanci u skladu sa opterećenjem. Sistem uspešno održava stabilnost u broju radnika kada je popunjenost reda između 30% i 70%, što je u skladu sa ciljevima.
- Performanse servera: Performanse servera tokom testiranja izgledaju zadovoljavajuće, s obzirom na to da je moguće obraditi zahteve kako ih redosledom primaju, čak i kada je opterećenje sistema povećano.
- Skalabilnost sistema: Testiranje sa više klijenata ukazuje na neophodnost prilagođavanja broja Worker-a u skladu sa povećanjem opterećenja, što sugeriše na efikasno rukovođenje resursima u cilju održavanja performansi sistema.

Objašnjenje odstupanja:

- Lošiji rezultati: Ukoliko su rezultati lošiji od očekivanih, mogući razlozi mogu uključivati nedovoljno optimizovan kod, nedostatak resursa na serveru ili potrebu za poboljšanjem algoritama dinamičkog balansiranja opterećenja.
- Bolji rezultati: Ako su rezultati bolji od očekivanih, to može ukazivati na dobru implementaciju i efikasno upravljanje resursima, ali istovremeno postavlja pitanje moguće potrebe za skalabilnošću sistema u realnom svetu.

Potencijalna unapređenja

- Optimizacija koda: Analiza performansi koda može pomoći u identifikaciji delova koji mogu biti dodatno optimizovani radi postizanja boljih rezultata.
- Dinamičko podešavanje parametara: Unapređenje sistema može uključivati dinamičko podešavanje parametara poput minimalnog i maksimalnog broja Worker-a u skladu sa stvarnim promenama u opterećenju sistema.
- Implementacija dodatnih metrika: Dodavanje dodatnih metrika za praćenje performansi sistema može pružiti dublji uvid u rad sistema i omogućiti preciznije prilagođavanje resursa.
- Testiranje pod ekstremnim opterećenjem: Proširenje testiranja pod ekstremnim uslovima može pomoći u identifikaciji slabosti sistema.
- Praćenje grešaka i logovanje: Uvođenje sistema praćenja grešaka i detaljnog logovanja može olakšati dijagnostiku i rešavanje problema u realnom vremenu.