D O K U M E N T A C I J A

Memory Management

Predmet:

Industrijski komunikaconi protokoli u ees

Grupa 33:

Bogdan Drljević PR 106/2021  
Vojin Velimirović PR 96/2021

**SADRŽAJ:**

1. Uvod
2. Dizajn
3. Strukture podataka
4. Rezultati testiranja
5. Zaključak
6. Potencijalna unapređenja

**Uvod**

## Opis problema koji se rešava

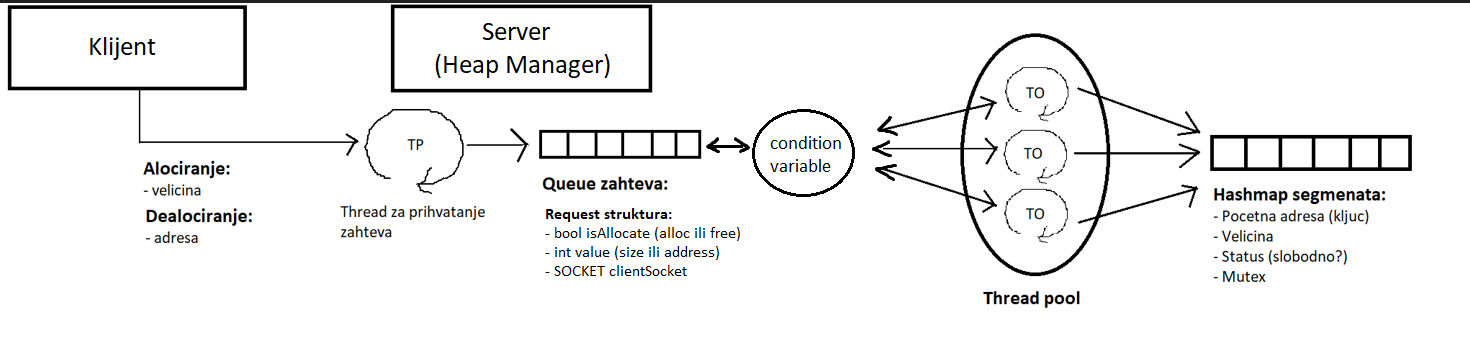
Heap Memory Manager je sistem za alokaciju i dealokaciju blokova memorije. Cilj sistema je omogućiti klijentima izdavanje komandi za zauzimanje i oslobađanje blokova memorije različitih veličina. Server koji obrađuje te zahtjeve mora ih obrađivati na način da se sistemski resursi, poput procesorskog vremena i potrošnje memorije, minimalno upotrebljavaju, istovremeno omogućavajući brz odziv na zadate komande.

## Ciljevi zadatka

* Omogućiti klijentima da zauzimaju i oslobađaju memoriju.
* Implementirati serversku aplikaciju koja efikasno obrađuje zahtjeve klijenata.
* Organizovati slobodnu memoriju na način da potrošnja memorije odgovara potrebama sistema.
* Heap manager mora biti dizajniran za rad u okruženju sa više niti.
* Implementirati strategiju za izbegavanje zagušenja u okruženju sa više niti, kako

bi se osigurale optimalne performanse i efikasno korišćenje memorije.

**Dizajn**



Arhitekturu možemo podijeliti na dva dijela: klijent i server (*heap manager*).

Klijent šalje zahtjeve serveru, server odgovara sa povratnom informacijom.

Zahtjevi su sljedećeg formata:

(<string, string>) <command\_code, value>

* command\_code – Označava operaciju koja treba da se izvrši.   
   Može biti 1 – allocate\_memory ili 2 – free\_memory
* value – U zavisnosti od *command\_code* parametra tumači se kao *size* ili *address.*

Server prima zahtjeve i stavlja ih u red. Po dodavanju zahtjeva u red spavajuće niti se probude i kreću da izvršavaju zahtjeve, sve dok red zahtjeva nije prazan. Zbog većeg broja zahtjeva i mogućnosti paralelnog izvršavanja ovdje koristimo thread pool.

Memorija na kojoj se vrše operacije je običan niz *Segment* stukture. Da bi se stanje niza pratilo koristimo Hashmapu segmenata koja čuva početne adrese i veličine zauzetih blokova.

Svaki segment sadrži svoj mutex, tako da više niti ne može istovremeno pristupi istim segmentima.

Ovo omogućava da više niti može pristupiti strukturi u isto vrijeme, ako bismo napravili mutex na nivou čitave strukture, ne bismo imali mogućnost paralelnog izvršavanja. Threadpool bi bio beskoristan i došlo bi do zagušenja.

Kao pomoćna struktura je implementirana uvezana lista *(eng. linked list)* slobodnih segmenata. Ovo ubrzava rad first fit algoritma, jer više ne mora da iterira kroz niz segmenata element po element. Lista slobodnih segmenata sadrži adresu početnog slobodnog segmenta u nizu slobodnih segmenata i broj slobodnih segmenata (tog niza) do sljedećeg zauzetog segmenta. Samim tim lista slobodnih segmenata ima znatno manje elemenata od niza segmenata, pa je iteriranje kroz nju kraće, u najboljoj situaciji *O(1)*.

Za traženje slobodnih segmenata se koristi first fit algoritam.

**Strukture podataka**

## Razlozi zbog kojih su izabrane baš te strukture podataka

Za segmente je izabran niz zbog minimalnog vremena pristupa. Možemo direktno da pristupimo nekoj adresi korišćenjem indeksa, što znači da prilikom zauzimanja ili oslobađanja memorije pristupamo segmentima brzinom *O(1).*

Za praćenje stanja niza segmenata i bilježenje zauzetih blokova memorije koristi se hashmapa zbog minimalnog vremena pristupa. Početna adresa se koristi kao ključ sa kojim se pristupa bloku.

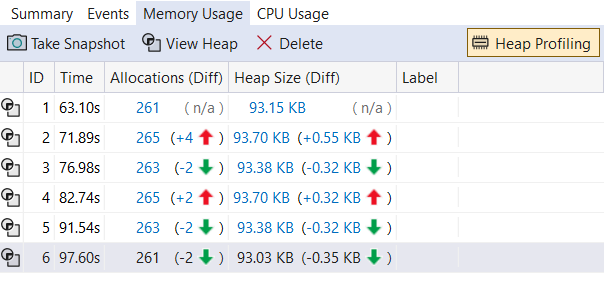
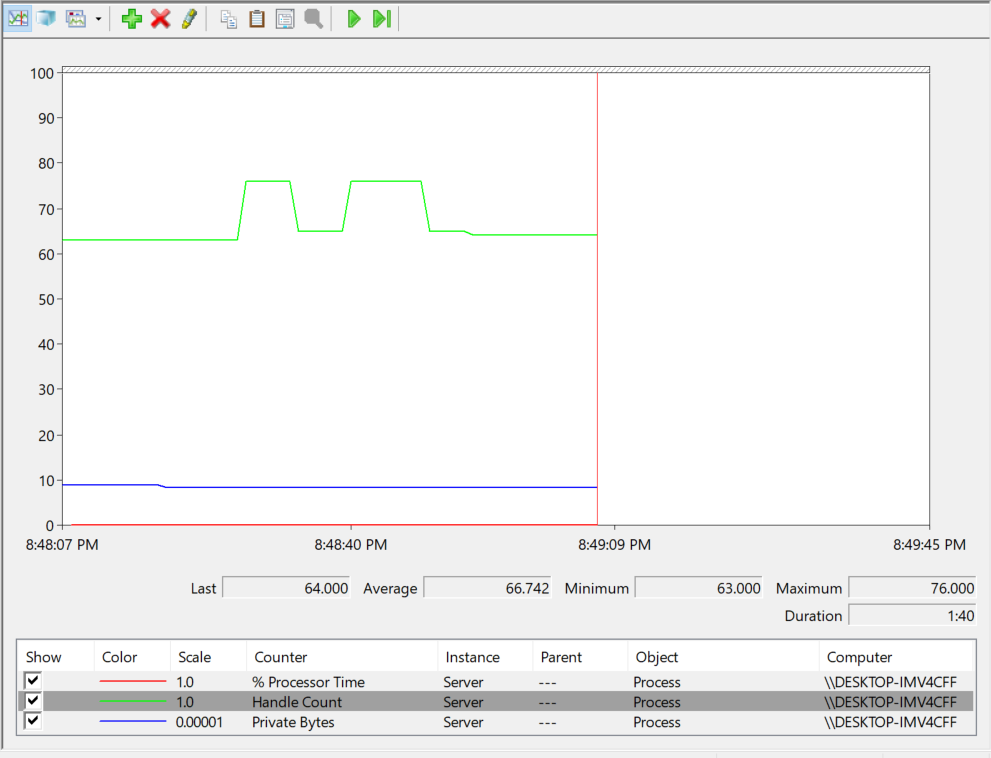
## Opis i pojašnjenje semantike podataka koje sadrže strukture

**Rezultati testiranja**

## Opis testova. Pojašnjenje zašto su odabrani baš ti testovi

Kao stres test je izabrano mnogo istovremenih zahtjeva od klijenta ka serveru. Pošto je jedan od zahtjeva da sistem mora da funckioniše u višenitnom okruženju i da se izbori sa zagušenjem, ovaj test će testirati otpornost sistema.

## Prikazani rezultati testova (tabele, dijagrami)

Testiranje memorije: Performance monitor: 

**Zaključak**

## Navesti zaključke koji proizilaze iz rezultata testova

## Objasniti zašto su rezultati lošiji/bolji od očekivanih

**Potencijalna unapređenja**

## Opis unapređenja koja je moguće uvesti na osnovu onoga što je navedeno u zaključku

## Navesti i objasniti razloge zbog kojih su predložena potencijalna unapređenja