Server de mesagerie

1. proiect realizat de:

**Matheiu Flavius**

**Pescaru Alexandru**

**Tutelea Dan**

2. Instalare

Pentru rulare, este nevoie doar de mașina virtuala de java, nu sunt necesare biblioteci suplimentare

3. Descrierea proiectului

Proiectul nostru este o aplicație care ofera un server de mesagerie care accepta transmiterea de mesage spre un destinatar sau publicare de mesage pe un anumit topic cu facilitatea de a extrage toate mesagele pe un anumit topic.

Structura proiectului:

3.1 serverul de topicuri

Server – partea de topic server

- acesta retien mesajele cu topic-uri într-o lista și returneaza toate mesajele cu topicu-ul cerut

- functia addTopic este folosită pentru a adauga un mesaj în coada. Aceasta mai are și rolul de a seta modifica timpul de stergere dacă acesta este mult mai mare decât timpul maxim al serverului.

-functia getTopic este folosită pentru a primi o lista de stringuri, reprezentând corpul mesajelor publicate cu un anumit topic-uri

- functia verifyTime decrementeaza timpul de stergere, și în caz ca acesta este 0, sterge mesajul din lista de mesaje de tipul topic

- functia run apeleaza la interval de 1 secunda functia verifyTime

TopicGenerator

- clasa folosita pentru generarea unor topicuri

- contine o lista de topicuri, de tipul String

- contine metoda get prin care returneaza un topic random din lista de topicuri

- aceasta garantează ca postarile și cererile de topicuri cer aceleași topicuri, pentru a nu apărea situații când se cer topicuri care nu vor fi postate sau invers

TopicAddClient

- clasa care reprezinta un client care adauga topicuri pe server

- extinde clasa Thread, iar in metoda run genereaza un String random de 15 caractere format din literele a-z, ii asociaza un timp de valabilitate si un topic luat random din lista de topicuri existenta. Apoi adauga noul Topic pe server, utilizand metoda addTopic a serverului

- metoda run are un delay de 1-3 secunde intre 2 postari consecutive, creat prin metoda sleep a thread-ului

TopicGetClient

- clasa care reprezinta un client care citeste topicuri de pe server

- extinde clasa Thread, iar in metoda run ia un topic random de pe server si afiseaza mesajul acelui topic preluat

- metoda run are un delay de 1-3 secunde pentru citirea unui topic, creat prin metoda sleep a thread-ului

3.2 Queue Server

Server – partea de queue server

-metoda addQueue adauga un mesaj in coada de mesaje daca mai exista loc, limita maxima aleasa este de 100.

-metoda getFromQueue scoate un mesaj din coada pentru clientul cu id-ul dat ca parametru

ClientIDMgr

-clasa pentru managementul clientiilor de tip ClientQueue construita dupa tiparul singleton pentru a asigura existenta unei singure instante pentru toti clientii

-conține o lista de întregi cu id-urile clienților

-are o metoda de a adaugă un id în lista de id-uri

-metoda getID returnează un id aleator și diferit de cel dat ca parametru pentru a evita situația în care un client și-ar putea trimite mesaj lui însuși.

ClientQueue

-clasa care reprezinta clienții care accesează coada din server

-metoda run trimite sau primește un mesaj spre, respectiv dinspre server în funcție de un flag modificat cu metoda random din clasa Math

3.3 Alte clase și funcții

QueueMessage și TopicMessage

- clase care funcționează ca niște structuri de date, care nu prezintă decât un corp fără funcționalitate

Main

- clasa care conține funcția main() și deci firul principal de execuția

- are doar rolul de a crea, porni (start) și de a aștepta finalizarea (join) firelor de execuție

Corectitudinea programului concurent se garantează prin utilizarea blocurilor syncronized și a variabilelor volatile, acolo unde am considerat ca sunt necesare.

Situațiile de bottleneck au fost eliminte prin proiectare, astfel:

- în cazul topic serverului, nu exista probleme de blocare. Adaugarea unui topic message este pur și simplu o adaugare în lista. Funcția getTopic parcurge lista, formeaza o noua lista pe care o retuneaza. În caz ca nu exista mesaje pe un anumit topic, lista va fi goala.

- în cazul queue message mecanismul este puțin mai „complicat”. Serverul dacă poate sa adauge un mesaj, îl adaugă și returnează *true*, altfel returneaza *false*. Clientul verifica valoare returnata, dacă e *false*, acesta reține mesajul și încearcă sa îl retransmită mai târziu. Scoaterea din lista este simpla, dacă primul mesaj din lista are id-ul destinatarului egal cu id-ul apelantului, se returnează mesajul, altfel *null.*