DOCUMENTATIE

TEMA 2

NUME STUDENT: SIMINA DAN-MARIUS
GRUPA: 30222

CUPRINS

1. Obiectivul temei

Obiectiv principal:

- Proiectarea și implementarea unei aplicații care vizează analiza sistemelor bazate pe cozi prin:
  1) Simularea unui șir de N clienți care sosesc pentru serviciu, intră în Q cozi, așteaptă, sunt serviți și în cele din urmă părăsesc cozile.
  2) Calcularea timpului mediu de așteptare, timpului mediu de serviciu și orei de vârf.

Sub-obiective:

1) Analiza problemei și identificarea cerințelor.
2) Proiectarea aplicației de simulare.
3) Implementarea aplicației de simulare.
4) Testarea aplicației de simulare.

2. Analiza problemei, modelare, scenarii, cazuri de utilizare

Cerințe funcționale:

- Aplicația de simulare ar trebui să permită utilizatorilor să configureze simularea.
- Aplicația de simulare ar trebui să permită utilizatorilor să înceapă simularea.

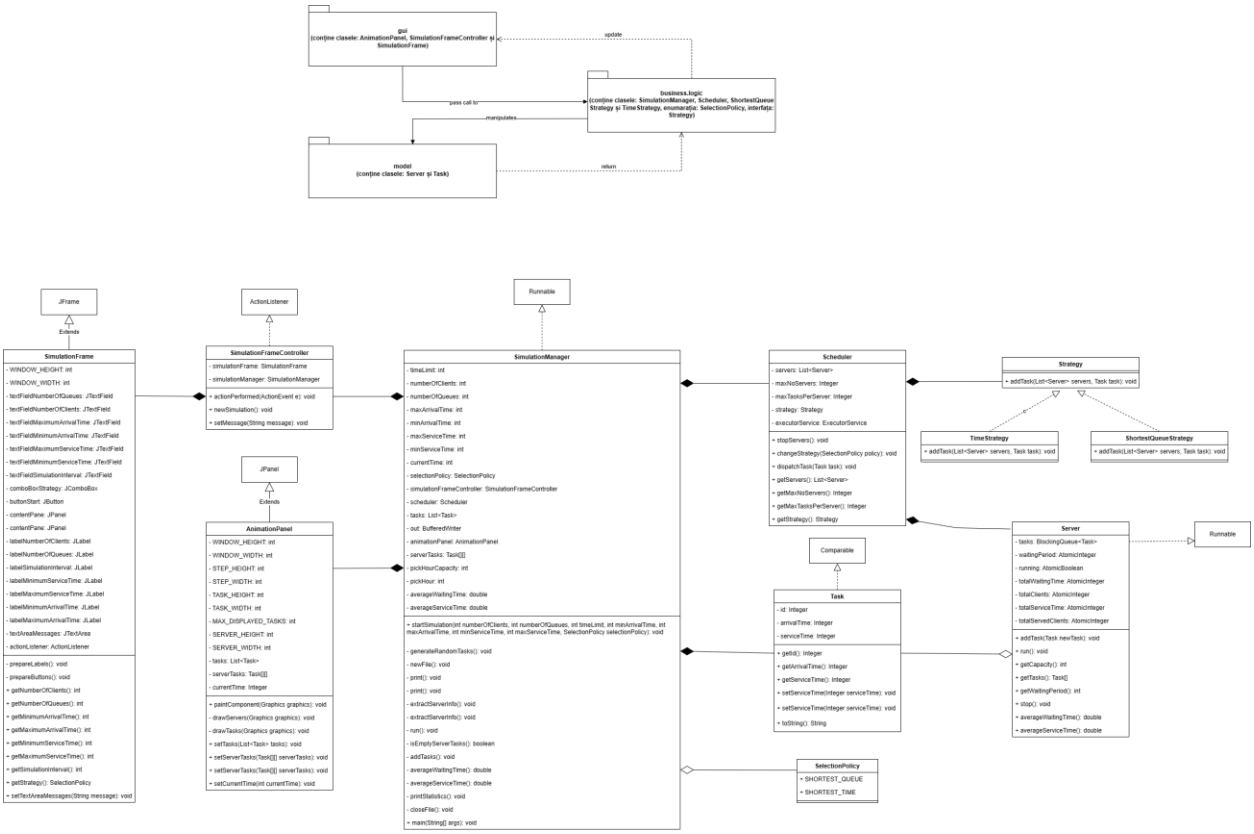
Cerințe non-funcționale:

- Aplicația de simulare ar trebui să fie intuitivă și ușor de utilizat de către utilizator.
- Aplicația de simulare ar trebui să afișeze cozile în timp real.
- Aplicația de simulare ar trebui să verifice datele introduse de la tastatură și să semnaleze utilizatorul dacă datele introduse nu sunt valide.

Scenarii:

- Scenariu de succes principal:
  1) Utilizatorul introduce valorile pentru:
     - Numărul de clienți,
     - Numărul de cozi,
     - Intervalul de simulare,
     - Timpul minim și maxim de sosire,
     - Timpul minim și maxim de serviciu.
  2) Utilizatorul apasă butonul de validare a datelor de intrare.
  3) Aplicația validează datele și afișează un mesaj care informează utilizatorul că a început simularea.
- Secvență alternativă: Valori nevalide pentru parametrii de configurare
  - Utilizatorul introduce valori nevalide pentru parametrii de configurare ai aplicației.
  - Aplicația afișează un mesaj de eroare și solicită utilizatorului să introducă valori valide.
  - Scenariul revine la pasul 1.

3. Proiectare



Aplicația e împarțită în 3 pachete model, gui și bussines.logic:
• Pachetul view conține clasele **AnimationPanel**, **SimulationFrameController** și **SimulationFrame** care implementează interfața grafică.
• Pachetul model conține clasa **Task** care reprezintă datele procesate de sever și clasa **Server** care implementeaza partea de simulare a procesării task-urilor.
• Pachetul bussines.logic conține clasele: SimulationManager care are rolul de a orchestra simularea, Scheduler care crează serverele și distribuie task-urile pe servere în functie de strategia aleasă, ShortestQueueStrategy și TimeStrategy repreintă strategiile puse la dispoziție, enumarația: SelectionPolicy, interfața: Strategy.

4. Implementare

Clasa **Task**:

- Modelează sarcina primită de server, conține un câmp **id** pentru identificare unica, campul **arrivalTime** care reprezintă timpul de sosire și câmpul **serviceTime** care reprezintă durata procesării.
- Metoda **setServiceTime** modifică câmpul care conține durata procesării.

Clasa **Server**:

- Simulează procesarea task-urilor, task-urile care așteaptă să fie procesate sunt stocate într-un BlockingQueue **tasks** care e sigură în ceea ce privește lucrul cu thread-uri, iar timpul de așteptare e stocat într-o variabila de tipul AtomicInteger **waitingPeriod** care și ea e sigură pentru thread-uri. Totodată mai sunt câmpuri care ajută la calcularea diferitor date statistice.

- Metoda **addTask** are rolul de a adăuga un task primit ca parametru în coadă, și incrementează timpul de așteptare.
- Metoda **run** simulează procesarea unui task, task-ul e extras din coadă apoi se așteaptă o secundă, dupa care durata pentru procesare e decrementă, totodată se actualizează perioada de așteptare și se modifică datele pentru calcularea statisticilor.
- Metoda **stop** are rolul de a semnala oprirea din execuție.
- Clasa mai conține metode pentru a obține valorile câmpurilor și diferite date statistice.

Clasa **Scheduler**:
- Conține lista de obiecte de tip **Server** și asignează fiecărui obiect un thread.
- Metoda **changeStrategy** modifică metoda pentru adăugare a task-urilor în server.
- Metoda **dispatchTask** în funcție de strategia selectată apelează metoda pentru a adăuga un task primit ca parametru într-un server convenabil.

Clasa **ShortestQueueStrategy**:
- Implementează metoda **addTask** care adaugă un task primit ca parametru în serverul care are cele mai puține task-uri în coadă, dacă există mai multe servere cu numar minim de task-uri în coadă se adaugă in primul.

Clasa **TimeStrategy**:
- Implementează metoda **addTask** care adaugă un task primit ca parametru în serverul care are cel mai mic timp de așteptare, dacă există mai multe servere cu timpul de așteptare minim se adaugă in primul.

Clasa **SimulationManager**:
- Metoda **generateRandomTasks** generează task-uri cu valori aleatoare în funcție de datele introduse de la tastatură.
- Metoda **newFile** crează și deschide un fișier nou pentru a scrie rezultatele în el.
- Metoda **print** scrie în fișier lista de task-uri si cozile din servere.
- Metoda **extractServerInfo** preia cozile din servere.
- Metoda **isEmptyServerTasks** verifică dacă există servere care au task-uri în coadă.
- Metoda **addTasks** trimite task-urile care au timpul de sosire egal cu timpul curent să fie adăugate într-un server.
- Metodele **averageWaitingTime** și **averageServiceTime** iau din fiecare server datele despre statistică și fac o medie pe servere.
- Metoda **printStatistics** afișează pe ecran și scrie în fișier datele statistice.
- Metoda **closeFile** închide fișierul.
- Metoda **startSimulation** primește datele pentru simulare reinițializează scheduler-ul, resetează timpul, generează task-uri noi și repornește simularea.
- Metoda **run** orchestrează simularea.

Clasa **AnimationPanel**:
- Metodele **setTasks**, **setServerTasks** și **setCurrentTime** setează datele despre simulare.
- Metodele **paintComponent**, **drawServers** și **drawTasks** generează animația în funcție de datele de simulare.

Clasa **SimulationFrameController**:
- Metoda **actionPerformed** este declanșată la apasarea butonului pentru începerea simularii și apelează mai departe funcțiile pentru începerea simularii.
- Metoda **setMessage** afișează mesaje în interfața grafică.
- Metoda **newSimulation** extrage datele introduse de la tastatură, dacă sunt corecte apelează **startSimulation** din **SimulationManager**, iar daca nu apelează **setMessage** pentru a trimite mesaj de eroare.

Clasa **SimulationFrame**:
- Conține metode pentru inițializarea elementelor grafice, obținerea datelor introduse și afișarea de messaje.

5. Rezultate

Test 1:

- N = 4
- Q = 2
- $T_{simulation}$ = 60
- $T_{min\_arrival}$ = 2
- $T_{max\_arrival}$ = 30
- $T_{min\_service}$ = 2
- $T_{max\_service}$ = 4

Rezultate – SHORTEST_QUEUE:

- Average waiting time: 0.0
- Pick hour: 21
- Average service time: 2.666666666666667

Rezultate – TIME_STRATEGY:

- Average waiting time: 0.0
- Pick hour: 21
- Average service time: 2.666666666666667

Test 2:

- N = 50
- Q = 5
- $T_{simulation}$ = 60
- $T_{min\_arrival}$ = 2
- $T_{max\_arrival}$ = 40
- $T_{min\_service}$ = 1
- $T_{max\_service}$ = 7

Rezultate – SHORTEST_QUEUE:

- Average waiting time: 1.6004545454545458
- Pick hour: 39
- Average service time: 4.073181818181818

Rezultate – TIME_STRATEGY:

- Average waiting time: 0.36380952380952375
- Pick hour: 6
- Average service time: 3.525555555555555

Test 3:

- N = 1000
- Q = 20
- $T_{simulation}$ = 200
- $T_{min\_arrival}$ = 10
- $T_{max\_arrival}$ = 100
- $T_{min\_service}$ = 3
- $T_{max\_service}$ = 9

Rezultate – SHORTEST_QUEUE:

- Average waiting time: 90.52214680573084
- Pick hour: 99
- Average service time: 5.544656185262805

Rezultate – TIME_STRATEGY:

- Average waiting time: 89.69342355838657
- Pick hour: 99
- Average service time: 5.517989403540427

6. Concluzii

În urma acestei teme am învățat cum funcționează thread-urile și cum pot fi folosite pentru a eficientiza prin executarea codului în paralel.

7. Bibliografie

ASSIGNMENT 2 – SUPPORT PRESENTATION
ASSIGNMENT 2 – SUPPORT PRESENTATION (PART II)
BlockingQueue (Java Platform SE 8) (oracle.com)
AtomicInteger (Java Platform SE 7) (oracle.com)
Stack Overflow - Where Developers Learn, Share, & Build Careers
GeeksforGeeks | A computer science portal for geeks