Modeliranje računalniških omrežij

2. laboratorijska naloga

Navodilo naloge

V orodju OMNeT++ razširite model M/M/1 sistema na M/M/c sistem (c predstavlja število strežnikov). Pri tem naj bo c podan kot parameter.

Namigi za reševanje

1. Opravila, ki so trenutno v strežbi, shranite v seznam (std::list):
Deklaracija seznama (v datoteki myQueue.h):

```
typedef std::list<cMessage *> jobsProcessingList;
jobsProcessingList jobsProcessing;
```

Dostopanje do elementov seznama:

```
jobsProcessingList::iterator msgIt;
for (msgIt=jobsProcessing.begin(); msgIt!=jobsProcessing.end(); msgIt++)
            {
                  (*msgIt)->...
                  ...
            }
```

Ob inicializaciji seznam izpraznite (`jobsProcessing.clear()`). V destruktorju izbrišite in prekličite vsa opravila v seznamu (`cancelAndDelete(*msg)`) - to so vsa opravila, ki so trenutno v strežbi.

2. V metodi **`handleMessage`** lahko naletite na dva tipa sporočil, in sicer na sporočila, ki jih je modul poslal samemu sebi in mu povejo, da je konec strežbe nekega opravila in sporočila, ki predstavljajo opravila in pridejo od zunaj skozi vhodna vrata. Sporočila bomo med seboj ločili glede na njihovo lastnost **`kind`**. Ločimo sledeči 2 možnosti:

2.1 Dobili smo sporočilo od zunaj, ki predstavlja neko opravilo `(msg->getKind() == 0)` Preverimo, če je na voljo kakšna strežna enota (ali je dolžina seznama krajša od števila strežnikov - c)

Odgovor DA: opravilo damo v strežbo - dodamo ga v seznam opravil (`jobsProcessing.push_back(msg)`), nastavimo njegovo lastnost Kind, preko katere bomo kasneje ugotovili, da je sporočilo lastno (`msg->setKind(10)`) in sporočilo pošljemo samemu sebi po času, ki je enak času strežbe (`scheduleAt( simTime() +serviceTime, msg)`)

Odgovor NE: preverimo, če je v čakalni vrsti prostor. Če ga ni, opravilo izbrišemo (`delete msg`). Če prostor je, opravilo vstavimo v čakalno vrsto (`queue.insert( msg )`).

2.2 Dobili smo lastno sporočilo, ki smo si ga prej poslali preko metode `scheduleAt` (`msg->getKind() == 10`). To sporočilo pomeni, da je strežba opravila končana, torej ga moramo poslati skozi izhodna vrata. Postopek:

- Sporočilo oziroma opravilo izbrišemo iz seznama (`jobsProcessing.erase(msgIt)`).

Sporočilo v seznamu najdemo glede na njegov ID (`if ((*msgIt)->getId()== msg->getId())`).

- Sporočilo oziroma opravilo pošljemo skozi izhodna vrata (`send(msg, "out")`).
- Preverimo, če je čakalna vrsta prazna. Če ni prazna, iz nje vzamemo naslednje opravilo, ki čaka na strežbo in mu dodelimo strežno enoto:

```
msg = check_and_cast<cMessage *>(queue.pop());
msg->setKind(10);
jobsProcessing.push_back(msg);
scheduleAt( simTime()+serviceTime, msg );
```