University of Ljubljana
Faculty of Computer and Information Science

Modeliranje računalniških omrežij

Uvod v laboratorijske vaje

OMNeT++
Moduli, Omrežja in Modeli

Torek, 15. oktobra 2013

Osnovni podatki

Prof. Miha Mraz (miha.mraz@fri.uni-lj.si)
As. Mattia Petroni (mattia.petroni@fri.uni-lj.si)

Spletna učilnica: https://ucilnica.fri.uni-lj.si

- Forum
- Obvestila
- Literatura
- Navodila za izdelavo seminarskih nalog

Pogoji za opravljanje predmeta (pravila igre in ocenjevanje)

- 50% sprotno delo na vajah
  - uspešno opravljen 1. seminar – 15% (poročilo + zagovor = pogoj za opravljanje 2. seminarja)
    IN
  - uspešno opravljen 2. seminar – 35 % (v skupini) (sprotno-tedensko poročanje + poročilo + zagovor)

- 50% izpit
  - pisni izpit
  - ustni izpit

Vaje

- V prvih 4 ciklih: primeri uporabe orodja **OMNeT++** in izdelava prvega seminarja
- V nadaljevanju: KONZULTACIJE – poročanje o napredku pri izdelavi drugega seminarja (v skupini)

Seminarji

- **1. seminar**: modeliranje in simuliranje osnovnih omrežij z uporabo osnovnih konceptov Teorije Strežbe in osnovnih modulov v knjižnicah orodja OMNeT++/INET.

- **2. seminar**: modeliranje in simuliranje kompleksnejšega (realnega) omrežja in podrobna performančna analiza omrežja.

Zagovor prvega seminarja

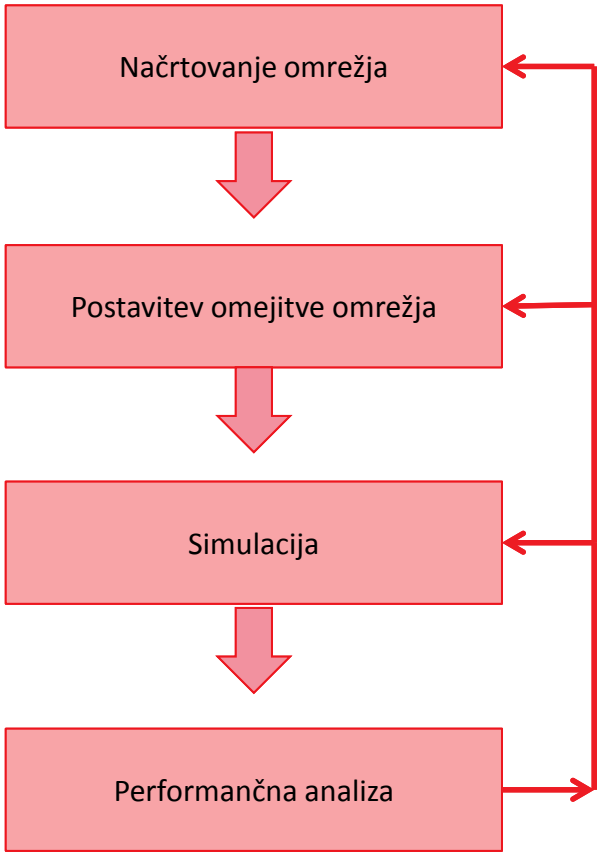
- **Obvezna oddaja poročila na spletni učilnici** v formatu PDF. Poročila naj bodo izdelana v orodju Latex! Hint: uporabljaj orodje Texmaker ali TeXnicCenter in prevajalnik PDFLatex.

- **Kratek zagovor na vajah.** Komentiranje rezultatov, preverjanje pravilnosti modela, etc.

Zagovor drugega seminarja

- **Obvezna oddaja tedenskega poročila v času konzultacijah.**
- Obvezna oddaja končnega poročila na spletni učilnici.
- **Kratka (max 10 minut) predstavitev rezultatov seminarja na vajah.**

Modeliranje računalniških omrežij



OMNeT++

- Je orodje za modeliranje računalniških omrežij.
- Omogoča modeliranje protokolov.
- Omogoča enostavno modeliranje zelo obsežnih omrežij (angl. *large-scale networks*).
- NI simulator temveč ponuja infrastrukturo za pisanje in poganjanje simulacij.
- IDE + GUI za poganjanje simulacij.
- Vsebuje konfigurabilno simulacijsko ogrodje *(simulation framework)*.
- Je objektno orientirano *(object oriented).*
- Temelji na diskretnih dogodkih *(discrete events).*
- Omogoča modularno modeliranje.

OMNeT++

- Programiranje
  - NED (glej *User Manual*, poglavje 3)
  - C++

OMNeT++

- Dokumentacija: http://www.omnetpp.org/documentation

- User Manual: http://www.omnetpp.org/doc/omnetpp/Manual.pdf
- IDE User Guide: http://www.omnetpp.org/doc/omnetpp/UserGuide.pdf
- Starting with IDE: http://www.omnetpp.org/doc/omnetpp/IDE-Overview.pdf
- Hitri uvod: http://www.omnetpp.org/pmwiki/index.php?n=Main.OmnetppInNutshell
- Enostavni primer: http://www.omnetpp.org/doc/omnetpp/tictoc-tutorial/

Moduli

- dva tipa modulov: glede na gnezdenje (*nesting)*
  - enostavni *(simple)* moduli
  - sestavljeni *(compound)* moduli

- enostavne module
  - programiramo sami (C++; OMNeT++ *simulation class library*)
  - obstajajo že napisani moduli (različne knjižnjice, zgledi)

Povezovanje modulov

- povezovanje preko vrat *(gates)*
  - vhodne, izhodne in dvosmerne
  - povezovanje vrat preko povezav *(connections)*

- komunikacija preko sporočil (*message passing)*

Opisovanje modulov (NED)

Z jezikom *NED* definiramo:

- paket *(package)*
- **ime modula**
- **tip modula** (*compound* ali *simple*)
- tipe (*types)*
- **parametre modula** (*parameters*)
- **vrata** *(gates: in, out, in/out)*
- **podmodule** (*submodules*): samo pri modulih, ki niso *simple*
- **povezave** (*connections*): med podmoduli

Opisovanje modulov (NED)

```
package ime_paketa;
module/simple ime_modula    // module = compound
                            // simple = brez podmodulov
{
    types:
        // tipi
    parameters:
        // parametri
    gates:
        // vrata, ki so lahko in, out ali inout
    submodules:
        // samo pri compound modulih
        modul1: TipPodmodula;
        modul2: TipPodmodula;
        ...
    connections:
        // povezave med podmoduli - samo pri compound
}
```

Opisovanje modulov (C++)

- določimo delovanje modula
- samo za preproste *(simple)* module
- programiranje na podlagi dogokov *(events)*

- podrobneje na naslednjih vajah

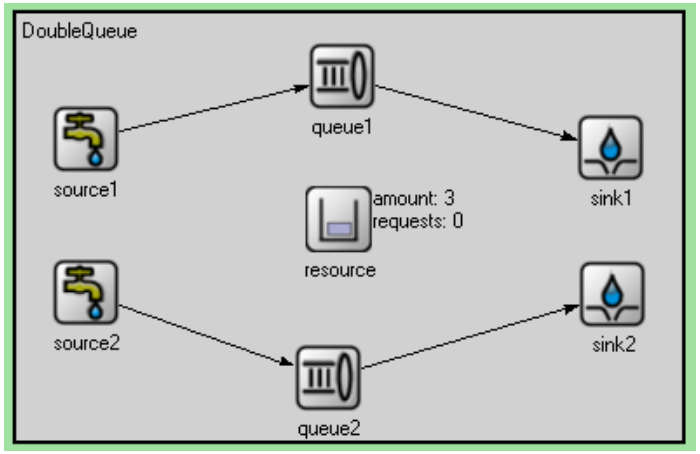
Model omrežja v OMNeT++

- predstavljen z omrežji (*networks*)
- omrežja so sestavljena iz modulov in povezav med njimi
- model opisujemo na dveh nivojih
  - v jeziku *NED (ned datoteke)*
  - v datoteki *omnetpp.ini*

- Z dvonivojskim opisovanjem ločimo model od eksperimentov
  - na *ned* datoteko gledamo kot na del modela
  - na *ini* datoteko gledamo kot na definicije eksperimentov

Model omrežja v OMNeT++

Vrednosti parametrov, ki so definirane v *ned* datotekah ne morejo biti povožene z vrednostmi v *ini* datotekah!



Opisovanje modela omrežja (NED)

Z jezikom *NED* definiramo posamezno omrežje *(network)*, in sicer:

- uporabo zunanjih modulov (*import*)
- **ime omrežja**
- tipi (*types*)
- **parametre** *(parameters)*
- **podmodule** *(submodules)*
- **povezave med podmoduli** *(connections)*
- lastnosti *(properties)* - @ime_lastnosti

Zgled: M/M/1 sistem - SimpleQueue.ned

```
import org.omnetpp.queueing.Queue;

import org.omnetpp.queueing.Sink;
import org.omnetpp.queueing.Source;


//
// This simple queueing network only contains a source, a FIFO queue
and a sink.
//
network SimpleQueue
{
    parameters:
        @display("i=block/network2");
    submodules:
        sink: Sink {
            @display("p=273,101");
        }
        queue: Queue {
            @display("p=165.0,79.0");
        }
        source: Source {
            @display("p=50.0,79.0");
        }
    connections:
        source.out --> queue.in++;
        queue.out --> sink.in++;
}
```

Opisovanje modela omrežja

```
import ... // za zunanje module (niso v istem projektu)

network Network1
{
    types:
    // tipi
    parameters:
    // parametri omrežja
    submodules:
    // moduli, ki nastopajo v omrežju
    connections:
    // povezave med podmoduli
}

network Network2
{
    ...
}
```

Opisovanje modela (omnetpp.ini)

- lahko vsebuje več omrežij
- omnetpp.ini je razdeljen na dva dela:
  - `[General]`: tu lahko določamo trajanje simulacije (`sim-time-limit`), določamo omrežje (če je samo eno)
  - `[Config imeKonfiguracije]`: določa posamezno konfiguracijo oziroma omrežje – lahko jih je več

Zgled (omnetpp.ini)

```
[General]

// ce imamo samo eno omrezje
  network = ImeOmrezja
  // parametri


// ce imamo vec omrezji

[Config ImeKonfiguracije1]
  description = "opis konfiguracije 1"
  network = ImeOmrezja1
  // parametri

[Config ImeKonfiguracije2]
  description = "opis konfiguracije 2"
  network = ImeOmrezja2
  // parametri
```

Tipi (*Types*)

- deklariramo jih lahko v sestavljenih modulih in omrežjih
- vidni so samo znotraj komponente
- definirajo tipe kanalov *(channels)* in modulov

```
types:
   channel mojKanal extends ned.DatarateChannel
   { datarate = 100Mbps; }
```

Parametri

- v C++ izvorni kodi preprostih modulov lahko določene spremenljivke/konstante vežemo na parametre

```
x = par("imeParametra");
```

- določanje vrednosti parametrov:
  - v *ned* datoteki modula,
  - v *ned* datoteki omrežja,
  - v datoteki *omentpp.ini* (če vrednost ni določena v *ned* datotekah)
  - ob zagonu simulacije (če vrednost ni določena nikjer)

Določanje parametrov (NED)

```
podatkovni_tip ime [@unit(enota – npr. s ali byte)]
               [@enum("moznost_1",…, "moznost_n")]
               [= default(vrednost/izraz)];
```

- `default`: lahko priredimo vrednost, izraz ali verjetnostno porazdelitev:
  - http://www.omnetpp.org/doc/omnetpp/api/group__RandomNumbers.html
- v primeru da želimo, da se izraz ovrednoti vsakič, ko simulacija potrebuje določen parameter uporabimo `volatile` parameter – primer: `volatile double serviceTime`

Določanje parametrov (omnetpp.ini)

```
omrezje.modul.parameter = vrednost/izraz
**.modul.parameter = vrednost/izraz
**.parameter = vrednost/izraz
```

Wildcards: `*,**,?,{a-f},…`

- le tisti parametri, ki še niso bili ovrednoteni v *ned* datotekah
- več ponovitev z različnimi parametrskimi vrednostmi

```
*.ime_parametra = ${1, 2, 5, 10..50 step 10}
*.ime_parametra = ${N=1, 2, 5, 10..50 step 10}
```

- več ponovitev z omejitvami

```
*.param1 = ${i=1..10 step 2}
*.param2 = ${j=1..20 step 3}
constraint = $j <= sqrt($i)
```

- spreminjanje *seed*-a s parametrom `seed-set` (privzeta vrednost je `${runnumber}`).

Vrata *(Gates)*

- povezujejo module z zunanjim svetom
- `input, output, inout`
- vektorski tip: `[], sizeof`

```
gates:
    input in;
    output out1[6];
    output out2[], out3[param1];
    inout io;
```

Povezave (*Connections*)

- povezave med vrati
- `input --> output;`
- `inout1 <--> inout2;`
- vektorski tip: `a++ --> b; // a je vektorski tip`
- uporaba kanalov
  - `IdealChannel` (default)
  - `DelayChannel` (zakasnitve)
  - `DatarateChannel` (datarate)

```
connections:
   x <--> {delay=10ms;} <--> z++;
   i <--> {datarate=100Mbps;} <--> j;
   a++ --> mojKanal --> b;
   a++ --> c;
```

Zaganjanje simulacije

- Run → Run As → OMNeT++ Simulation
- [izbira konfiguracije]
- Run/Fast Run/Express Run/Run Until
- Stop/Call finish()

Parametri

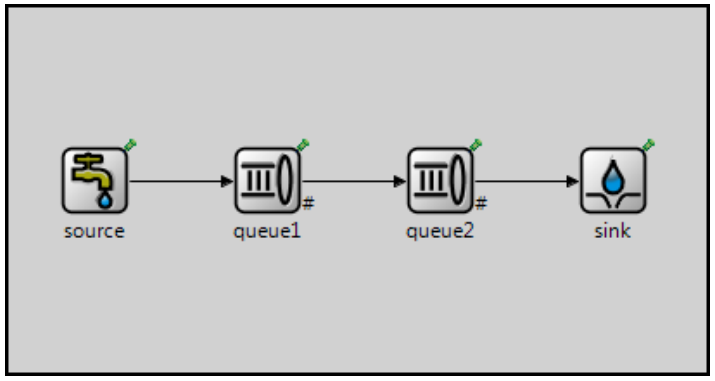
- v C++ izvorni kodi preprostih modulov lahko določene spremenljivke/konstante vežemo na parametre

```
x = par("imeParametra");
```

- določanje vrednosti parametrov:
  - v *ned* datoteki modula,
  - v *ned* datoteki omrežja,
  - v datoteki *omentpp.ini* (če vrednost ni določena v *ned* datotekah)
  - ob zagonu simulacije (če vrednost ni določena nikjer)

Primer

- V novem projektu implementirajte omrežje na spodnji sliki. Pri tem uporabite module projekta *queinglib*. Vsi kanali naj bodo idealni.



- Poženite sledeče simulacije
  - `queue1 serviceTime`: spreminja se od `3s do 10s`
  - `queue2 serviceTime`: spreminja se od `1s do 2s`
  - `interArrivalTime`: porazdeljen normalno, `mean` se spreminja od `1s` do `10s`, `stddev` pa je vedno enak `mean*0.5`