



Universiti Malaysia Sabah

Faculty of Computing and Informatics

KP00303

NETWORK SIMULATION

SEM 1-2023/2024

ASSIGNMENT 2

NAME	ID
SELVA GANAPATHY A/L MUTHU KUMAR	BI20110217
YOGANATHAN SHUNMUGAM	BI20160312
PAGITHEREN A/L UMABATHY	BI20160316
NESHADRAJ A/L SASIDHARAN	BI20160350

KP00303 Network Simulation

Faculty of Computing and Informatics, Universiti Malaysia Sabah

Semester 1, 2023/24

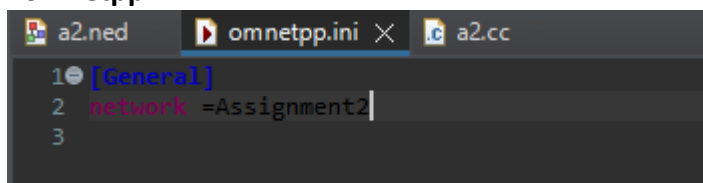
Lecturer: Shaliza Hayati A. Wahab

Assignment 2 (100 marks)

This assignment is a group of three persons. Due date: 11 January 2024, 5:00 pm. Submit all the answers and simulations codes in class website and the printout to my office (Room 65, Level 2, Block A).

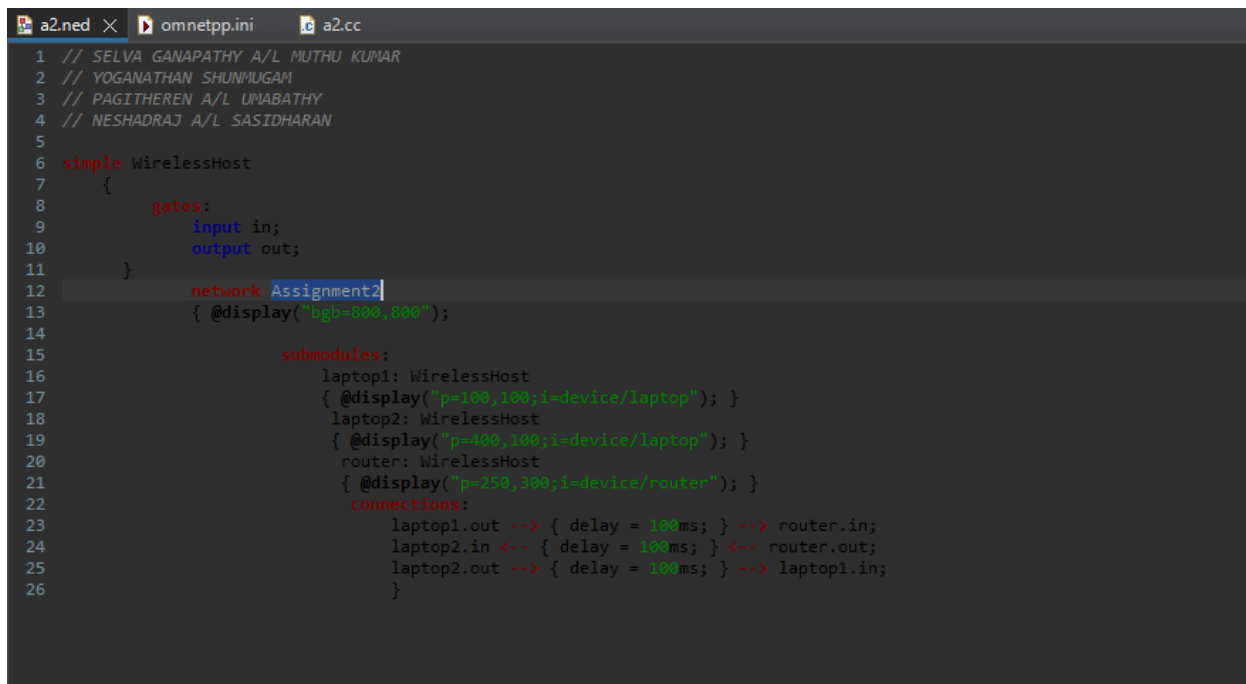
1. Create OMNeT++ simulation source codes (.ini, .ned and .cc) to test a wireless network with one wireless router and two laptops. Put your names and matric numbers in the coding.

omnetpp.ini:



```
a2.ned  omnetpp.ini x a2.cc
1 [General]
2 network =Assignment2
3
```

a2.ned:



```
a2.ned x omnetpp.ini a2.cc
1 // SELVA GANAPATHY A/L MUTHU KUMAR
2 // YOGANATHAN SHUNMUGAM
3 // PAGITHEREN A/L UMABATHY
4 // NESHADRAJ A/L SASIDHARAN
5
6 simple WirelessHost
7 {
8     gates:
9         input in;
10        output out;
11    }
12    network Assignment2
13    { @display("hgh=800,800");
14
15        submodules:
16            laptop1: WirelessHost
17            { @display("p=100,100,i=device/laptop"); }
18            laptop2: WirelessHost
19            { @display("p=400,100,i=device/laptop"); }
20            router: WirelessHost
21            { @display("p=250,300,i=device/router"); }
22        connections:
23            laptop1.out --> { delay = 100ms; } --> router.in;
24            laptop2.in <-- { delay = 100ms; } <-- router.out;
25            laptop2.out --> { delay = 100ms; } --> laptop1.in;
26    }
```

a2.cc:

```
a2.ned  omnetpp.ini  a2.cc X
1
2 // SELVA GANAPATHY A/L MUTHU KUMAR
3 // YOGANATHAN SHUNMUGAM
4 // PAGITHIREN A/L UNABATHY
5 // NESHADRAJ A/L SASIDHARAN
6
7
8 #include <stdio.h>
9 #include <string.h>
10 #include <omnetpp.h>
11
12 using namespace omnetpp;
13
14 class WirelessHost : public cSimpleModule{
15     private:
16         simtime_t timeout;
17         cMessage *timeoutEvent; int seq;
18         cMessage *message;
19
20     public:
21         WirelessHost();
22         virtual ~WirelessHost();
23     protected:
24         virtual cMessage *generateNewMessage();
25         virtual void sendCopyOf(cMessage *msg);
26         virtual void initialize() override;
27         virtual void handleMessage(cMessage *msg) override;
28 };
29 Define_Module(WirelessHost);
30 WirelessHost::WirelessHost(){
31     timeoutEvent = message = nullptr;
32 }
33 WirelessHost::~WirelessHost(){
34     cancelAndDelete(timeoutEvent); delete message;
35 }
```

```
a2.ned  omnetpp.ini  a2.cc X
35 }
36 void WirelessHost::initialize(){
37     seq = 0; timeout = 1.0;
38     timeoutEvent = new cMessage("timeoutEvent");
39     EV << "Sending initial message\n";
40     message = generateNewMessage(); sendCopyOf(message);
41     scheduleAt(simTime()+timeout, timeoutEvent);
42 }
43 void WirelessHost::handleMessage(cMessage *msg){
44     if (msg == timeoutEvent) {
45         EV << "Timeout expired, resending message and restarting timer\n";
46         sendCopyOf(message);
47         scheduleAt(simTime()+timeout, timeoutEvent);
48     }
49     else {
50         EV << "Received: " << msg->getName() << "\n";
51         delete msg; EV << "Timer cancelled.\n";
52         cancelEvent(timeoutEvent); delete message;
53         message = generateNewMessage();
54         sendCopyOf(message);
55         scheduleAt(simTime()+timeout, timeoutEvent);
56     }
57 }
58 cMessage *WirelessHost::generateNewMessage(){
59     char msgname[20];
60     sprintf(msgname, "msg-%d", ++seq);
61     cMessage *msg = new cMessage(msgname);
62     return msg;
63 }
64 void WirelessHost::sendCopyOf(cMessage *msg){
65     cMessage *copy = (cMessage *)msg->dup();
66     send(copy, "out");
67 }
68 }
```

Simulation:

OMNeT++/QtEnv (debug) - General #0 - omnetpp.ini - F:\omnetpp-6.0.2\LabsExe\wirelessnetwork

File Simulate Inspect View Help

next: #48 1s 600ms 000us 000ns 000ps

Next: msg-16 (omnetpp::cMessage, id=98) In: Assignment2.laptop2 (WirelessHost, id=3) At: 1.6s (now+0s)

Assignment2 (Assignment2) id=1
simulation.scheduled-events (cEventHeap) length=6

laptop1 (WirelessHost) id=2
laptop2 (WirelessHost) id=3
router (WirelessHost) id=4

Assignment2

laptop1 (WirelessHost) id=2
laptop2 (WirelessHost) id=3
router (WirelessHost) id=4

Zoom:1.00x

INFO: Received: msg-15
INFO: Timer cancelled.
** Event #45 t=1.5 Assignment2.router (WirelessHost, id=4) on msg-15 (omnetpp::cMessage, id=92)
INFO: Received: msg-15
INFO: Timer cancelled.
** Event #46 t=1.6 Assignment2.router (WirelessHost, id=4) on msg-16 (omnetpp::cMessage, id=94)

General #0: Assignment2 Msg stats: 6 scheduled / 9 existing / 103 created

Source code:

omnetpp.ini:

[General]

network = Assignment2

a2.ned:

```
// SELVA GANAPATHY A/L MUTHU KUMAR
// YOGANATHAN SHUNMUGAM
// PAGITHEREN A/L UMABATHY
// NESHADRAJ A/L SASIDHARAN

simple WirelessHost
{
    gates:
        input in;
        output out;
    }

    network Assignment2
    { @display("hgb=800,800");

        submodules:
            laptop1: WirelessHost
            { @display("p=100,100;i=device/laptop"); }
            laptop2: WirelessHost
            { @display("p=400,100;i=device/laptop"); }
            router: WirelessHost
            { @display("p=250,300;i=device/router"); }
        connections:
            laptop1.out --> { delay = 100ms; } --> router.in;
            laptop2.in <-- { delay = 100ms; } <-- router.out;
            laptop2.out --> { delay = 100ms; } --> laptop1.in;
        }
    }
```

a2.cc:

```
// SELVA GANAPATHY A/L MUTHU KUMAR
// YOGANATHAN SHUNMUGAM
// PAGITHEREN A/L UMABATHY
// NESHADRAJ A/L SASIDHARAN

#include <stdio.h>
#include <string.h>
#include <omnetpp.h>

using namespace omnetpp;

class WirelessHost : public cSimpleModule{
private:
    simtime_t timeout;
    cMessage *timeoutEvent; int seq;
    cMessage *message;

public:
    WirelessHost();
    virtual ~WirelessHost();

protected:
    virtual cMessage *generateNewMessage();
    virtual void sendCopyOf(cMessage *msg);
    virtual void initialize() override;
    virtual void handleMessage(cMessage *msg) override;
};

Define_Module(WirelessHost);
```

```

WirelessHost::WirelessHost(){
    timeoutEvent = message = nullptr;
}
WirelessHost::~WirelessHost(){
    cancelAndDelete(timeoutEvent); delete message;
}
void WirelessHost::initialize(){
    seq = 0; timeout = 1.0;
    timeoutEvent = new cMessage("timeoutEvent");
    EV << "Sending initial message\n";
    message = generateNewMessage(); sendCopyOf(message);
    scheduleAt(simTime()+timeout, timeoutEvent);
}
void WirelessHost::handleMessage(cMessage *msg){
    if (msg == timeoutEvent) {
        EV << "Timeout expired, resending message and restarting timer\n";
        sendCopyOf(message);
        scheduleAt(simTime()+timeout, timeoutEvent);
    }
    else {
        EV << "Received: " << msg->getName() << "\n";
        delete msg; EV << "Timer cancelled.\n";
        cancelEvent(timeoutEvent); delete message;
        message = generateNewMessage();
        sendCopyOf(message);
        scheduleAt(simTime()+timeout, timeoutEvent);
    }
}
cMessage *WirelessHost::generateNewMessage(){
    char msgname[20];
    sprintf(msgname, "msg-%d", ++seq);
    cMessage *msg = new cMessage(msgname);
    return msg;
}
void WirelessHost::sendCopyOf(cMessage *msg){
    cMessage *copy = (cMessage *)msg->dup();
    send(copy, "out");
}

```

2. Consider the following single-server queuing system from time = 0 to time = 25 sec.
Arrivals and service times are as follows:

- Customer 1 arrives at t = 1 second and requires 5 seconds of service time
- Customer 2 arrives at t = 1 second and requires 2 seconds of service time
- Customer 3 arrives at t = 2 seconds and requires 3 seconds of service time
- Customer 4 arrives at t = 12 seconds and requires 6 seconds of service time

Calculate the system throughput (X), total busy time (B), mean service time (T_s), utilization (U), mean system time (delay in system) (W), and mean number in the system (L).

$$\begin{aligned} \text{a) System throughput } (X) &= \frac{\text{Number of customer (completed job)}}{\text{Total time}} \\ &= \frac{4}{25} \\ &= 0.16 \end{aligned}$$

$$\begin{aligned} \text{b) Total busy time } (B) &= \text{Total service time for all job} \\ &= 5 + 2 + 3 + 6 \\ &= 16 \end{aligned}$$

$$\begin{aligned} \text{c) Mean service time } (T_s) &= \frac{\text{Total service time } (B)}{\text{Number customer (complete job)}} \\ &= \frac{16}{4} \\ &= 4 \end{aligned}$$

$$\begin{aligned} \text{d) Utilization } (U) &= \frac{\text{Total service time } (B)}{\text{Total time}} \\ &= \frac{16}{25} \\ &= 0.64 \end{aligned}$$

$$\begin{aligned} \text{e) Mean system time (delay in system) } (W) &= \frac{\text{Total time spend in the system (service time + waiting time)}}{\text{Number of customer (completed job)}} \\ &= \frac{[(1+0+5)+(1+5+2)+(2+7+3)+(12+13+6)]}{4} \\ &= 14.25 \end{aligned}$$

$$\begin{aligned} \text{f) Mean number in the system } (L) &= \text{Utilization } (U) \times \text{Mean system time } (W) \\ &= 0.64 \times 14.25 \\ &= 9.12 \end{aligned}$$

GitHub link:

[GitHub Repo](#)