Assignment 1

SYSC 5104

Mobile Communication Networks

Misagh Tavanpour

Department of System and Computer Engineering

Carleton University

October 2012.

**Part I**

Mobile networks or cellular networks are kind of [radio](http://en.wikipedia.org/wiki/Radio) networks distributed over land areas. Each of these land areas known as a cell. Each cell has at least one fixed [transceiver](http://en.wikipedia.org/wiki/Transceiver) called [base station](http://en.wikipedia.org/wiki/Base_station) and to avoid interference it uses a different set of frequencies from neighboring cells. These cells support radio coverage over a geographic area by joining together and this enables portable transceivers such as smart phones to communicate with each other via fixed transceivers (Fig 1). This communication can be hold even if both side of this communication are moving through more than one cell during transmission.



**Fig 1: A sample mobile network**

As it can be seen from Fig 1 in a mobile network a geographical area includes a number of cells which they themselves consist of base stations (BS) and user equipments (UE). For simplicity we consider Fig 2 as the base for conceptual model of mobile network. Fig 2 reveals that our geographical area consists from three cells and all of them have one BS. Both cell 1 and cell 2 have one UE and there are two active UE in Cell 3. We have assumed BSs are connected to each other through wired connection and a communication from or to other geographic areas will be hold via cell 3’s BS.



**Fig 2: A mobile network with three cells**

Fig 3 shows the conceptual model for this network. In this model both geographic area and cell are considered as coupled model whereas BS and UE are considered as atomic model. Geographic area as coupled model compose of three other coupled model (cells) which each of them consist of two types of atomic models (BS and UE). BSs can communicate with users and themselves. They maybe transmit a received packet for a UE in their cell or forward it to other BS. UE can communicate with each other through their base stations so they only can send/receive to/from base stations. A BS is in the passive mode in default. If it receives a packet it will switch to active mode and it will process the packet and generate the related output (forward the packet) and after that it will be back to passive mode again (until another packet reaches to it). According to its definition a BS can be defined with two states: Idle and Receive Packet (RecPack). In Idle state a BS will be on passive mode and waiting for an input. By receiving an input it will change the state and it will go to RecPack state. In the latter it will process the received packet to forward it to its destination. After that it will back to Idle mode again and this process will execute for all input packets. Fig 4 shows the state diagram of a BS according to DEVS Graph rules.

 **Fig 3: Proposed model for sample mobile network**

A UE can generate traffic packet for its destinations (act as a source of communication). Also when a UE receive a traffic packet as the final destination of the packet it should create an Ack message and send it back for the sender. A UE can be defined with four states: Idle, SendPack, RecPack and RecAck. Fig 5 illustrates the UE state diagram according to DEVS Graph rules.



**Fig 4: DEVS Graph schema of the Base Station (BS)**



**Fig 5: DEVS Graph schema of the User Equipment (UE)**

**Part II**

According to Fig 3 the atomic models are: UE1, UE2, UE3, UE4, BS3Port1, BS3Port2 and BS5Port1. All the UEs functionality is same (Fig 5). BS3Port1 and BS3Port2 are same too. As it was mentioned in Part I their differences with BS5Port are the responsibility of the BS5 to communicate with other areas and the number of user it deals with. But same as UEs all the BSs functionality is same too (Fig 4). The coupled models include Cells and in the top level an area. The detail of the connections between all the components and their hierarchically structure has been shown in Fig 3. In the following the formal specification of the atomic model and coupled models has been discussed. Also there are detailed descriptions of each functions responsibility.

**i) Atomic Models**

**UE =**

{

if (msg.port() == In )

{

Temp = msg.value();

Temp = Temp % 2;

if (Temp == 0) //This is a Data Packet.

{

state = RecPack;

Temp = msg.value()+1; //Ack for received packet.

}

else state = RecAck //This ia a Ack Pack from Receiver.

holdIn(Atomic::active, ProcessTime);

}

return \*this;

}

{

switch (state)

{

case Idle: //Prepare the initialization to

//generate next packet each GenerateTime.

//Ex:(src\*100000)+(des\*10000)+(SentPacketNum \* 2)

//Ex: First Pack from UE1 to UE2 🡺 120002

// Save odd numbers for Ack of pack:

// if PackNum = 120002 🡺 AckNum = 120003

//In this scenario consider src = 1 and des = 2

Temp = 120000 + (SentPacketNum \* 2);

state = SendPack;

holdIn( Atomic::active, GenerateTime);

break;

case RecPack:

ReceivedPacketNum ++;

// To keep number of Receive packets

state = Idle;

break;

case RecAck:

state = Idle;

break;

case SendPack:

SentPacketNum ++;

state = Idle;

break;

}

return \*this;

}

{

if (state == RecPack) //Send ACK to source of received packet.

sendOutput( msg.time(), Out, Temp) ;

else if (state == SendPack)

sendOutput( msg.time(), Out, Temp) ; //Send Packet

return \*this ;

}

**BS3Port =**

{

Temp = msg.value();

Temp = Temp % 2;

if (Temp == 0)

// This is Traffic Packet from Src to Des then forward it for Des

{ // Try to find the suitable output port.

Temp = msg.value();

PortName = (Temp / 10000) % 10;

//Ex: ((120110 / 10000) % 10) ==> (12 % 10) ==> 2

}

else //This is Ack Pack from Des to Src then forward it for Src

{

Temp = msg.value();

PortName = Temp / 100000; //Ex: 120111 / 100000 = 1

}

PackNum = msg.value();

state = RecPack;

holdIn(Atomic::active, ProcessTime);

return \*this;

}

{

if (state == RecPack)

{

state = Idle;

passivate();

}

return \*this;

}

// This Func is defined for BS3Port1 according to Fig 3.

// For the BS3Port2 we just need change the output port according

// to same Fig.

{

if (state == RecPack)

{

if (PortName == 1) sendOutput( msg.time(), Out1, PackNum) ;

else if (PortName == 2) sendOutput( msg.time(), Out3, PackNum) ;

else if ((PortName == 9) || (PortName == 3) || (PortName == 4))

sendOutput( msg.time(), Out2, PackNum) ;

}

return \*this ;

}

**BS5Port =**

{

Temp = msg.value();

Temp = Temp % 2;

if (Temp == 0)

// This is Traffic Packet from Src to Des then forward it for Des

{ // Try to find the suitable output port.

Temp = msg.value();

PortName = (Temp / 10000) % 10;

//Ex: ((120110 / 10000) % 10) ==> (12 % 10) ==> 2

}

else //This is Ack Pack from Des to Src then forward it for Src

{

Temp = msg.value();

PortName = Temp / 100000; //Ex: 120111 / 100000 = 1

}

PackNum = msg.value();

state = RecPack;

holdIn(Atomic::active, ProcessTime);

return \*this;

}

{

if (state == RecPack)

{

state = Idle;

passivate();

}

return \*this;

}

// This Func is defined for BS5Port1 according to Fig 3.

{

if (state == RecPack)

{

if (PortName == 1) sendOutput( msg.time(), Out5, PackNum);

else if (PortName == 2) sendOutput( msg.time(), Out4, PackNum);

else if (PortName == 3) sendOutput( msg.time(), Out1, PackNum);

else if (PortName == 4) sendOutput( msg.time(), Out2, PackNum);

else if (PortName == 9) sendOutput( msg.time(), Out3, PackNum);

}

return \*this ;

}

**ii) Coupled Models**

**Part III**

There are test bench for each atomic and coupled models in the attached file (Mobile.zip) to this report. Each of those components has a special event file (ComponentName.ev: UE1.ev, Cell3.ev and etc) which tries to check the correctness of its component. For each component this event file and its related outputs are in the folder with same name as the component. For example the related files for BS3Port1 are in: …/Mobile/BS3Port1.

For all UEs we need to test their reaction to received data packet (they should create Ack for it) and they response to received Ack packet (Just require to keep the number and there is no need for output). Also they should generate packet at each certain time. The results show that this component works properly.

The event file and the Output file for UE1 are like this (Same thing happened for other UEs. Their file included to the attached file.):

UE1.ev:

00:00:00:25 In 910000 🡺 **00:00:00:035 out 910001**

00:00:00:40 In 910002

00:00:01:15 In 910004

00:00:01:35 In 910006

00:00:01:55 In 910008

00:00:02:15 In 910010

00:00:02:35 In 910012

00:00:02:55 In 910014 🡺 **00:00:02:065 out 910015**

00:00:03:15 In 910016

00:00:03:35 In 910018

00:00:03:50 In 910020

**00:00:04:50 In 120051 🡺 00:00:04:060 out 25**

UE1out.out

**00:00:00:035 out 910001**

**00:00:00:050 out 910003**

00:00:00:100 out 120002

00:00:00:150 out 120004

00:00:00:200 out 120006

00:00:00:250 out 120008

00:00:00:300 out 120010

00:00:00:350 out 120012

00:00:00:400 out 120014

00:00:00:450 out 120016

00:00:00:500 out 120018

00:00:00:550 out 120020

00:00:00:600 out 120022

00:00:00:650 out 120024

00:00:00:700 out 120026

00:00:00:750 out 120028

00:00:00:800 out 120030

00:00:00:850 out 120032

00:00:00:900 out 120034

00:00:00:950 out 120036

00:00:01:000 out 120038

00:00:01:025 out 910005

00:00:01:045 out 910007

00:00:01:065 out 910009

00:00:01:115 out 120040

00:00:01:165 out 120042

00:00:01:215 out 120044

00:00:01:265 out 120046

00:00:01:315 out 120048

00:00:01:365 out 120050

00:00:01:415 out 120052

00:00:01:465 out 120054

00:00:01:515 out 120056

00:00:01:565 out 120058

00:00:01:615 out 120060

00:00:01:665 out 120062

00:00:01:715 out 120064

00:00:01:765 out 120066

00:00:01:815 out 120068

00:00:01:865 out 120070

00:00:01:915 out 120072

00:00:01:965 out 120074

**00:00:02:025 out 910011**

**00:00:02:045 out 910013**

**00:00:02:065 out 910015**

00:00:02:115 out 120076

00:00:02:165 out 120078

00:00:02:215 out 120080

00:00:02:265 out 120082

00:00:02:315 out 120084

00:00:02:365 out 120086

00:00:02:415 out 120088

00:00:02:465 out 120090

00:00:02:515 out 120092

00:00:02:565 out 120094

00:00:02:615 out 120096

00:00:02:665 out 120098

00:00:02:715 out 120100

00:00:02:765 out 120102

00:00:02:815 out 120104

00:00:02:865 out 120106

00:00:02:915 out 120108

00:00:02:965 out 120110

**00:00:03:025 out 910017**

**00:00:03:045 out 910019**

**00:00:03:060 out 910021**

00:00:03:110 out 120112

00:00:03:160 out 120114

00:00:03:210 out 120116

00:00:03:260 out 120118

00:00:03:310 out 120120

00:00:03:360 out 120122

00:00:03:410 out 120124

00:00:03:460 out 120126

00:00:03:510 out 120128

00:00:03:560 out 120130

00:00:03:610 out 120132

00:00:03:660 out 120134

00:00:03:710 out 120136

00:00:03:760 out 120138

00:00:03:810 out 120140

00:00:03:860 out 120142

00:00:03:910 out 120144

00:00:03:960 out 120146

00:00:04:010 out 120148

**00:00:04:060 out 25**

00:00:04:110 out 120150

00:00:04:160 out 120152

00:00:04:210 out 120154

00:00:04:260 out 120156

00:00:04:310 out 120158

00:00:04:360 out 120160

00:00:04:410 out 120162

00:00:04:460 out 120164

00:00:04:510 out 120166

00:00:04:560 out 120168

00:00:04:610 out 120170

00:00:04:660 out 120172

00:00:04:710 out 120174

00:00:04:760 out 120176

00:00:04:810 out 120178

00:00:04:860 out 120180

00:00:04:910 out 120182

00:00:04:960 out 120184

In case of BSs, they should be able to forward the received packet correctly and according to their test results they work properly as well. They receive packet and in 10 ms later they forward it for packet destination.

BS3Port1.ev:

00:00:00:25 In2 910000

00:00:00:40 In2 910002

00:00:01:35 In1 120006

00:00:01:55 In3 120009

00:00:02:15 In1 120016

00:00:02:35 In1 910013

00:00:03:15 In1 120018

00:00:03:35 In3 120019

00:00:03:50 In2 910020

BS3Port1out.out:

00:00:00:035 out1 910000

00:00:00:050 out1 910002

00:00:01:045 out3 120006

00:00:01:065 out1 120009

00:00:02:025 out3 120016

00:00:02:045 out2 910013

00:00:03:025 out3 120018

00:00:03:045 out1 120019

00:00:03:060 out1 910020

The output of other components are way large because UEs keep generating the packets and BSs forward those packages and then these packet may travel among the cells or even want to go to outside area. All the event and output files have been put in Mobile.zip in their category. Also the final part of the simulation with the entire components can be found in the following path of attached file: …/Mobile/Mobile.