METHODOLOGIES FOR DISCRETE EVENT MODELING AND SIMULATION
SYSC 5104

Assignment #1

Simulation of location update in GSM mobile network

Dan Liu

Part I: The Conceptual Model

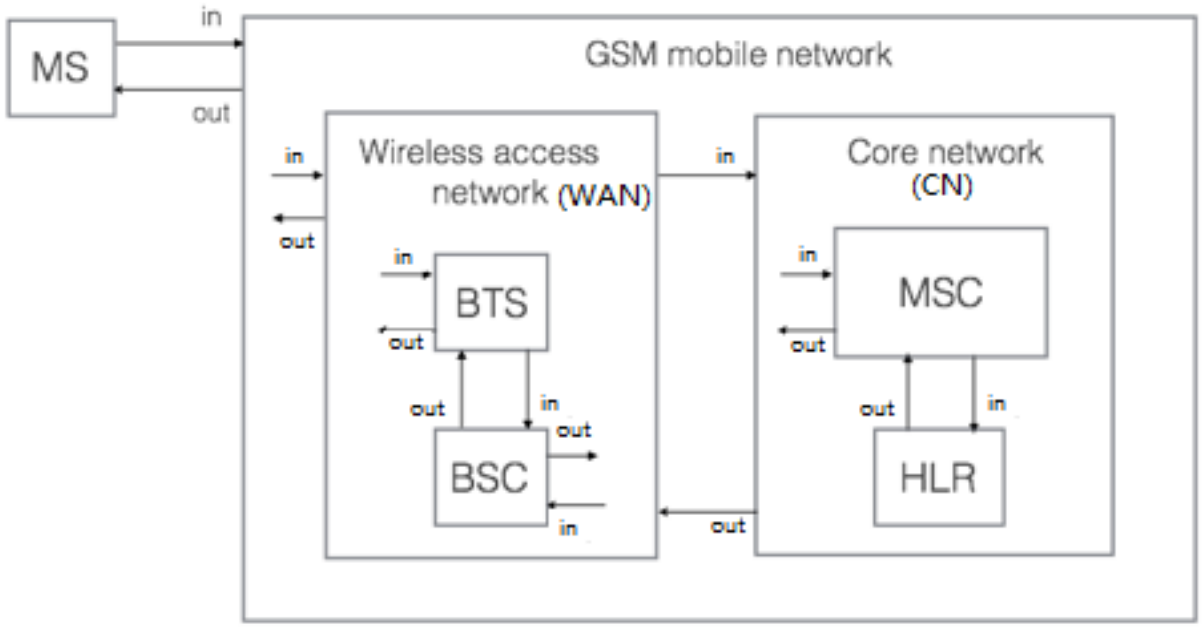
Description of Problem

Mobility is one of the most important functionality of GSM mobile network, which tracks where MS(mobile subscriber, such as cell phone) are and allows various network services to be delivered to them seamlessly in various locations. Location update is a crucial procedure to implement the functionality of mobility in the GSM mobile network.
Briefly, location update is a procedure of periodically updating the exact current position of mobile subscriber, commonly a location area in the GSM mobile network. With maintained latest location information of MS, the GSM mobile network is able to deliver various network services to them when required.

In this assignment, I plan to build a basic model of the GSM mobile network and then simulate the location update procedure in different scenarios.

Brief Sketch of Model Structure

First of all, I give out a brief sketch of the GSM mobile network where location update procedure is performed.



Description of Components

MS represents terminal clients of the GSM mobile network, such as cell phone and laptop.

As shown, basically, the GSM mobile network consists of 2 parts, Wireless access network and Core network.

1.Wireless access network(WAN)

MS connects to Wireless access network to access the GSM mobile network through wireless channel. MS sends and receives message to the GSM mobile network through Wireless access network.

To be more specific, there are 2 components in Wireless access network, BTS(Base Transceiver Station) and BSC(Base Station Controller). BTS is a outdoor station in charge of sending and receiving wireless signal while BSC is the controller part of Wireless access network.

2.Core network(CN)

Namely, Core network plays a core role in the GSM mobile network, which provides most of core functionalities of the network delivered to MS(such as call and SMS). A Core network can be much more complex than illustrated. I simplify its' structure which only involves related parts with location update procedure.

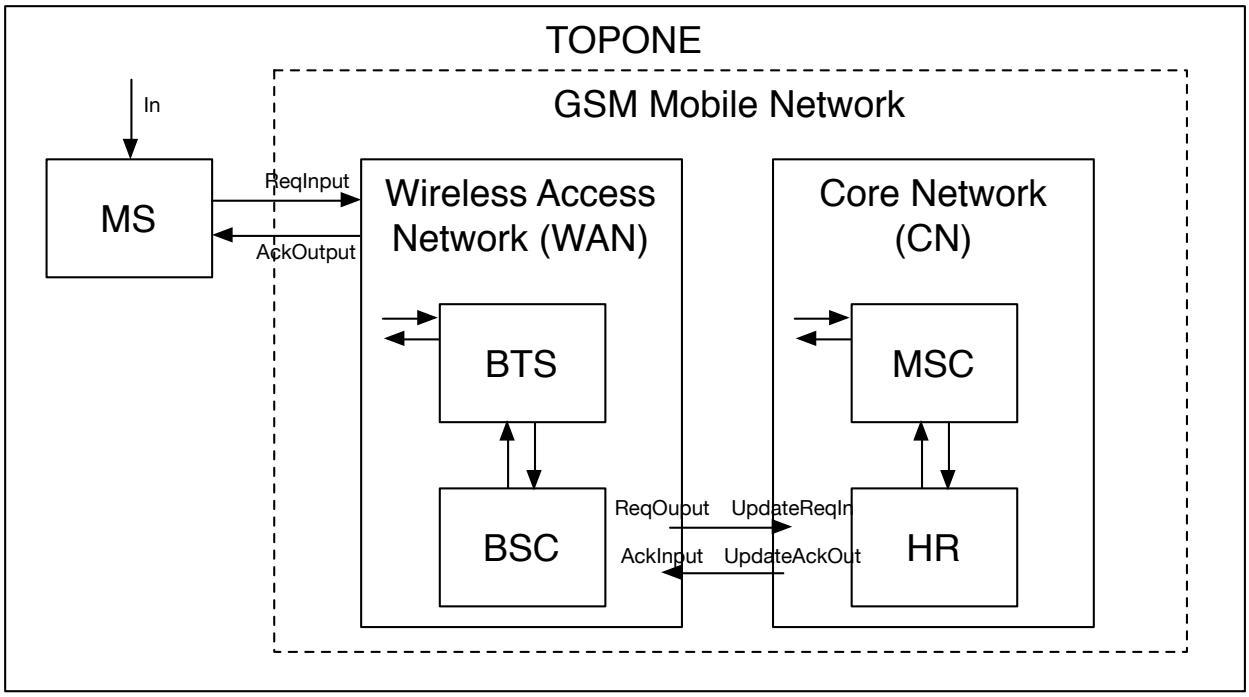
As shown, MSC(Mobile Switching Center) and HLR(Home Location Register) are 2 basic parts of Core network. MSC is one of the most important part of Core network, which implements the majority of business logics processing, including call switching, data routing, MS information storage and so on. Among them, MS information storage will be discussed detailedly in my assignment because location update procedure is just one application of it.

Besides, HLR is also very important part of Core network where lots of essential data of MS is stored. It plays a key role in location update procedure in some scenarios.

Part II: DEVS Formal Specification

As described in the part I, there are 3 levels in the entire simulated system which contains 5 atomic models totally: MS, BTS, BSC, MSC and HR. BTS and BSC are coupled to form WAN model while MSC and HR are coupled to form CN model. WAN and CN compose the complete GSM mobile network which provides mobile service to MS. MS, WAN and CN are coupled to form TOPONE model which simulates the entire location update procedure in GSM mobile network.

Next, formalism specification of atomic models and coupled models will be presented successively as illustrated in the following figure:



Formalism Specification of Atomic Models

1.MS

State Variables:
sigma = INFINITY, phase = IDLE; // { IDLE, UPDATE }
update_result = 0; // { DEFAULT, SUCC, FAIL }
imsi = XXX; // IMSI of MS
ta(IDLE) = INFINITY
ta(UPDATE) = 1min

X = { force_update, update_ack }
Y = { update_request }
S = { {sigma, phase, update_result, imsi} }

δext(S, e, X)
{
  case phase

    IDLE:
      if X is from force_update
        phase = UPDATE;

    UPDATE:
      if X is from update_ack
        update_result = SUCC;
        phase = IDLE;

```
}

δint( S, e )
{
   case phase

      IDLE:
         update_result = DEFAULT;
         paasivate();

      UPDATE:
        passivate();
}

λ( UPDATE )
{
   send update_request to out port;
}
```

2.BTS

```
X = { update_request_i, update_ack_i }
Y = { update_request_o, update_ack_o }
S = { IDLE, UPDATING, ACK }
ta(IDLE) = INFINITY
ta(UPDATING) = 10s
ta(ACK) = 10s


δext( S, e, X )
{
   case phase

      IDLE:
         if X is from update_request_i
            phase = UPDATING;

      ACK:
         if X is from update_ack_i
            phase = ACK;
}

δint( S, e, X )
{
   case phase
      IDLE:
```

```
        passivate();

    UPDATING:
        phase = ACK;
        passivate();

    ACK:
        phase = IDLE;
}

λ( UPDATING )
{
    send update_request to out port update_request_o;
}
λ( ACK )
{
    send update_ack to out port update_ack_o;
}
```

3.BSC

```
X = { update_request_i, update_ack_i }
Y = { update_request_o, update_ack_o }
S = { IDLE, UPDATING, ACK }


δext( S, e, X )
{
    case phase

    IDLE:
        if X is from update_request_i
            phase = UPDATING;
    ACK:
        if X is from update_ack_i
            phase = ACK;
}

δint( S, e )
{
    case phase
    IDLE:
        passivate();
    UPDATING:
        phase = ACK;
        passivate();
```

```
        ACK:
            phase = IDLE;
}

λ( UPDATING )
{
    send update_request to out port update_request_o;
}
λ( ACK )
{
    send update_ac to out port update_ack_o;
}
```

4.MSC

```
X = { update_request_i, hr_ack_i }
Y = { hr_request_o, update_ack_o }
S = { IDLE, UPDATING, ACK, REQ_HR }


δext( S, e, X )
{
    case phase

        IDLE:
            if X is from update_request_i
            {
              if ( isSameMSC )
              {
                phase = UPDATING;
              }
              else
              {
                phase = REQ_HR
              }
            }

        REQ_HR:
            if X is from hr_ack_i
                phase = ACK;
}

δint( S, e )
{
    case phase
        IDLE:
```

```
        passivate();
    UPDATING:
        phase = ACK;
        passivate();
    ACK:
        phase = IDLE;
    REQ_HR:
        passivate();
}

λ( REQ_HR )
{
    send hr_request to out port hr_request_o;
}
λ( ACK )
{
    send update_ack to out port update_ack_o;
}
```

5.HR

```
X = { hr_request_i }
Y = { hr_ack_o }
S = { IDLE, UPDATING, ACK }


δext( S, e, X )
{
    case phase

        idle:
            if x from hr_request_i
                phase = updating;

}

δint( S, e )
{
    case phase
        idle:
            passivate();
        Updating:
            phase = Ack;
        Ack:
            phase = idle;
}
```

λ(ACK)
{
  send hr_ack to out port hr_ack_o;
}

Formalism Specification of Coupled Models

1,WAN coupled model(BTS, BSC)

X = { ReqInput, AckInput }
Y = { ReqOutput, AckOutput }
D = { BTS, BSC }
I(BTS) = {BSC}
I(BSC) = {BTS}
Z(BTS) = BSC
Z(BSC) = BTS
SELECT: ({ BTS, BSC }) = BSC

2,CN coupled model(MSC, HR)

X = { UpdateReqIn }
Y = { UpdateAckOut }
D = { MSC, HR }
I(MSC) = {HR}
I(HR) = {MSC}
Z(MSC) = HR
Z(HR) = MSC
SELECT: ({MSC, HR}) = MSC

3,TOPONE coupled model(MS, WAN, CN)

X = { In }
Y = { NULL }
D = { MS, WAN, CN }
I(MS) = { WAN }
I(WAN) = { CN }
Z(MS) = WAN
Z(WAN) = CN
SELECT: ({MS, WAN, CN}) = CN

Test Strategy

Firstly, atomic models will be tested individually using different input events files(.ev) to assure the correctness of atomic models. Then coupled models(WAN, CN, TOPONE) will be tested integrating atomic models using different input events files in order to verify the correctness of entire simulation system.

Part III: Implementation and Results

Atomic models and coupled models defined in part II are implemented using CD++. Relevant source code files are packaged with this document.

After implementation, each atomic model and coupled model are tested individually using different input event files(.ev), generating corresponding output files which shows correct results as expected. Correct results as expected prove the correctness of models, including atomic models, coupled models and entire simulation system.

Analysis of execution and result of each models is presented as following successively.

1.MS

```
//define -100 - -1 as instructions contained in message
#define FAIL              -1
#define FORCE_UPDATE      -2
#define FORCE_UPDATE_ACK -3
#define UPDATE_REQ        -4
#define UPDATE_ACK        -5
```

As defined in source code file, I use some minus integers to represent instructions communicated between models which will be contained in input event files(.ev) to simulate input events in test.

For MS atomic model, It is assumed that a force_update_request is received every 30ms and then 15ms after this a update_ack is received, by defining input event sequence in .ev file as follow:

```
00:00:00:15 In -2
00:00:00:30 In -3
00:00:00:45 In -2
00:00:00:60 In -3
00:00:00:75 In -2
00:00:00:90 In -3
```

Posing above input event file, MS atomic model generated correct output results as expected shown in the follow:

```
00:00:00:025 out -4
00:00:00:055 out -4
00:00:00:085 out -4
```

The results shown above is consistent with expectation which could prove the correctness of this atomic model.

2.BTS

For BTS atomic model, It is assumed that a update_request is received every 30ms and then 15ms after this a update_ack is received, by defining input event sequence in .ev file as follow:

```
00:00:00:15 ReqInput -4
00:00:00:30 AckInput -5
00:00:00:45 ReqInput -4
00:00:00:60 AckInput -5
00:00:00:75 ReqInput -4
00:00:00:90 AckInput -5
```

Posing above input event file, BTS atomic model generated correct output results as expected shown in the follow:

```
00:00:00:025 reqoutput -4
00:00:00:040 ackoutput -5
00:00:00:055 reqoutput -4
00:00:00:070 ackoutput -5
00:00:00:085 reqoutput -4
00:00:00:100 ackoutput -5
```

The results shown above is consistent with expectation which could prove the correctness of this atomic model.

3.BSC

For BSC atomic model, It is assumed that a update_request is received every 30ms and then 15ms after this a update_ack is received, by defining input event sequence in .ev file as follow:

```
00:00:00:15 ReqInput -4
00:00:00:30 AckInput -5
00:00:00:45 ReqInput -4
00:00:00:60 AckInput -5
00:00:00:75 ReqInput -4
00:00:00:90 AckInput -5
```

Posing above input event file, BSC atomic model generated correct output results as expected shown in the follow:

```
00:00:00:025 reqoutput -4
00:00:00:040 ackoutput -5
00:00:00:055 reqoutput -4
00:00:00:070 ackoutput -5
00:00:00:085 reqoutput -4
00:00:00:100 ackoutput -5
```

The results shown above is consistent with expectation which could prove the correctness of this atomic model.

4.MSC

In the purpose of simplifying the stuff, for MSC atomic model, I only consider the most complex scenario that MS performs location update in different MSC scope which means MSC will interact with HR to update old information stored in HR.

In my simulation, It is assumed that a update_request is received by MSC every 30ms and then 15ms after this a hr_ack is received, by defining input event sequence in .ev file as follow:

```
00:00:00:15 UpdateReqIn -4
00:00:00:30 HRAckIn -7
00:00:00:45 UpdateReqIn -4
00:00:00:60 HRAckIn -7
00:00:00:75 UpdateReqIn -4
00:00:00:90 HRAckIn -7
```

Posing above input event file, MSC atomic model generated correct output results as expected shown in the follow:

```
00:00:00:015 hrreqout -6
00:00:00:040 updateackout -5
00:00:00:045 hrreqout -6
00:00:00:070 updateackout -5
00:00:00:075 hrreqout -6
00:00:00:100 updateackout -5
```

The results shown above is consistent with expectation which could prove the correctness of this atomic model.

5.HR

For HR atomic model, It is assumed that a hr_request is received every 30ms, by defining input event sequence in .ev file as follow:

```
00:00:00:15 In -6
00:00:00:45 In -6
00:00:00:75 In -6
00:00:00:105 In -6
00:00:00:135 In -6
00:00:00:165 In -6
```

Posing above input event file, HR atomic model generated correct output results as expected shown in the follow:

```
00:00:00:035 out -7
00:00:00:065 out -7
00:00:00:095 out -7
00:00:00:125 out -7
00:00:00:155 out -7
00:00:00:185 out -7
```

The results shown above is consistent with expectation which could prove the correctness of this atomic model.

6.WAN coupled model

For WAN coupled model, It is assumed that a update_request is received every 90ms and then 45ms after this a update_ack is received, by defining input event sequence in .ev file as follow:

```
00:00:00:15 ReqInput -4
00:00:00:60 AckInput -5
00:00:00:105 ReqInput -4
00:00:00:150 AckInput -5
00:00:00:195 ReqInput -4
00:00:00:240 AckInput -5
```

Posing above input event file, WAN coupled model generated correct output results as expected shown in the follow:

```
00:00:00:035 reqoutput -4
00:00:00:080 ackoutput -5
00:00:00:125 reqoutput -4
00:00:00:170 ackoutput -5
00:00:00:215 reqoutput -4
00:00:00:260 ackoutput -5
```

The results shown above is consistent with expectation which could prove the correctness of this coupled model.

7.CN coupled model

For CN coupled model, It is assumed that a update_request is received every 60ms, by defining input event sequence in .ev file as follow:

```
00:00:00:15 UpdateReqIn -4
00:00:00:75 UpdateReqIn -4
00:00:00:135 UpdateReqIn -4
```

Posing above input event file, CN coupled model generated correct output results as expected shown in the follow:

```
00:00:00:045 updateackout -5
00:00:00:105 updateackout -5
00:00:00:165 updateackout -5
```

The results shown above is consistent with expectation which could prove the correctness of this coupled model.

8.TOPONE coupled model

For TOPONE coupled model, It is assumed that a force_update_request is received every 1s, by defining input event sequence in .ev file as follow:

```
00:00:00:15 In -2
00:00:01:15 In -2
00:00:02:15 In -2
```

As predicted, there is no output generated after simulation, posing above input event file. However, we can check the console output to track the execution of simulation which is shown as follow:

```
MSC initFunction()
HR initFunction()
BTS initFunction()
BSC initFunction()
MS initFunction()
MSC outputFunction() at 00:00:00:000
MSC internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
HR outputFunction() at 00:00:00:000
HR internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
BTS outputFunction() at 00:00:00:000
BTS internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
BSC outputFunction() at 00:00:00:000
BSC internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MS outputFunction() at 00:00:00:000
MS internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MS externalFunction() at 00:00:00:015, value: -2
                I received FORCE_UPDATE ext message -2   at 00:00:00:01
```

```
MS outputFunction() at 00:00:00:025
                I sent update request with imsi 123456   at 00:00:00:025
MS internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is UPDATE
BTS externalFunction() at 00:00:00:025, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:00:025
BTS outputFunction() at 00:00:00:035
                I sent update request at 00:00:00:035
BTS internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is ACK
BSC externalFunction() at 00:00:00:035, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:00:035
BSC outputFunction() at 00:00:00:045
                I sent update request at 00:00:00:045
BSC internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is ACK
MSC externalFunction() at 00:00:00:045, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:00:045
MSC outputFunction() at 00:00:00:045
                I sent HR request at 00:00:00:045
MSC internalFunction()
                The input state is REQ_HR
                Call passivate()
                The Output state is REQ_HR
HR externalFunction() at 00:00:00:045, value: -6
                I received HR_REQ ext message -6   at 00:00:00:045
HR outputFunction() at 00:00:00:055
HR internalFunction()
                The input state is UPDATE
                The Output state is ACK_UPDATE
HR outputFunction() at 00:00:00:065
                I sent update ack at 00:00:00:065
HR internalFunction()
                The input state is ACK_UPDATE
                The Output state is IDLE
MSC externalFunction() at 00:00:00:065, value: -7
                I received HR_ACK ext message -7   at 00:00:00:065
HR outputFunction() at 00:00:00:065
HR internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MSC outputFunction() at 00:00:00:075
                I sent update ack at 00:00:00:075
MSC internalFunction()
                The input state is ACK_UPDATE
                The Output state is IDLE
BSC externalFunction() at 00:00:00:075, value: -5
                I received UPDATE_ACK ext message -5   at 00:00:00:075
MSC outputFunction() at 00:00:00:075
```

```
MSC internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
BSC outputFunction() at 00:00:00:085
                I sent update ack at 00:00:00:085
BSC internalFunction()
                The input state is ACK
                The Output state is IDLE
BTS externalFunction() at 00:00:00:085, value: -5
                I received UPDATE_ACK ext message -5   at 00:00:00:085
BSC outputFunction() at 00:00:00:085
BSC internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
BTS outputFunction() at 00:00:00:095
                I sent update ack at 00:00:00:095
BTS internalFunction()
                The input state is ACK
                The Output state is IDLE
MS externalFunction() at 00:00:00:095, value: -5
                I received FORCE_UPDATE_ACK ext message -5   at 00:00:00:095
BTS outputFunction() at 00:00:00:095
BTS internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MS outputFunction() at 00:00:00:105
MS internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MS externalFunction() at 00:00:01:015, value: -2
                I received FORCE_UPDATE ext message -2   at 00:00:01:015
MS outputFunction() at 00:00:01:025
                I sent update request with imsi 123456   at 00:00:01:025
MS internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is UPDATE
BTS externalFunction() at 00:00:01:025, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:01:025
BTS outputFunction() at 00:00:01:035
                I sent update request at 00:00:01:035
BTS internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is ACK
BSC externalFunction() at 00:00:01:035, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:01:035
BSC outputFunction() at 00:00:01:045
                I sent update request at 00:00:01:045
BSC internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is ACK
MSC externalFunction() at 00:00:01:045, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:01:045
```

```
MSC outputFunction() at 00:00:01:045
                I sent HR request at 00:00:01:045
MSC internalFunction()
                The input state is REQ_HR
                Call passivate()
                The Output state is REQ_HR
HR externalFunction() at 00:00:01:045, value: -6
                I received HR_REQ ext message -6   at 00:00:01:045
HR outputFunction() at 00:00:01:055
HR internalFunction()
                The input state is UPDATE
                The Output state is ACK_UPDATE
HR outputFunction() at 00:00:01:065
                I sent update ack at 00:00:01:065
HR internalFunction()
                The input state is ACK_UPDATE
                The Output state is IDLE
MSC externalFunction() at 00:00:01:065, value: -7
                I received HR_ACK ext message -7   at 00:00:01:065
HR outputFunction() at 00:00:01:065
HR internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MSC outputFunction() at 00:00:01:075
                I sent update ack at 00:00:01:075
MSC internalFunction()
                The input state is ACK_UPDATE
                The Output state is IDLE
BSC externalFunction() at 00:00:01:075, value: -5
                I received UPDATE_ACK ext message -5   at 00:00:01:075
MSC outputFunction() at 00:00:01:075
MSC internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
BSC outputFunction() at 00:00:01:085
                I sent update ack at 00:00:01:085
BSC internalFunction()
                The input state is ACK
                The Output state is IDLE
BTS externalFunction() at 00:00:01:085, value: -5
                I received UPDATE_ACK ext message -5   at 00:00:01:085
BSC outputFunction() at 00:00:01:085
BSC internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
BTS outputFunction() at 00:00:01:095
                I sent update ack at 00:00:01:095
BTS internalFunction()
                The input state is ACK
                The Output state is IDLE
MS externalFunction() at 00:00:01:095, value: -5
                I received FORCE_UPDATE_ACK ext message -5   at 00:00:01:095
BTS outputFunction() at 00:00:01:095
BTS internalFunction()
                The input state is IDLE
                Call passivate()
```

```
                The Output state is IDLE
MS outputFunction() at 00:00:01:105
MS internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MS externalFunction() at 00:00:02:015, value: -2
                I received FORCE_UPDATE ext message -2   at 00:00:02:015
MS outputFunction() at 00:00:02:025
                I sent update request with imsi 123456   at 00:00:02:025
MS internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is UPDATE
BTS externalFunction() at 00:00:02:025, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:02:025
BTS outputFunction() at 00:00:02:035
                I sent update request at 00:00:02:035
BTS internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is ACK
BSC externalFunction() at 00:00:02:035, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:02:035
BSC outputFunction() at 00:00:02:045
                I sent update request at 00:00:02:045
BSC internalFunction()
                The input state is UPDATE
                Call passivate()
                The Output state is ACK
MSC externalFunction() at 00:00:02:045, value: -4
                I received UPDATE_REQ ext message -4   at 00:00:02:045
MSC outputFunction() at 00:00:02:045
                I sent HR request at 00:00:02:045
MSC internalFunction()
                The input state is REQ_HR
                Call passivate()
                The Output state is REQ_HR
HR externalFunction() at 00:00:02:045, value: -6
                I received HR_REQ ext message -6   at 00:00:02:045
HR outputFunction() at 00:00:02:055
HR internalFunction()
                The input state is UPDATE
                The Output state is ACK_UPDATE
HR outputFunction() at 00:00:02:065
                I sent update ack at 00:00:02:065
HR internalFunction()
                The input state is ACK_UPDATE
                The Output state is IDLE
MSC externalFunction() at 00:00:02:065, value: -7
                I received HR_ACK ext message -7   at 00:00:02:065
HR outputFunction() at 00:00:02:065
HR internalFunction()
                The input state is IDLE
                Call passivate()
                The Output state is IDLE
MSC outputFunction() at 00:00:02:075
                I sent update ack at 00:00:02:075
```

```
MSC internalFunction()
                    The input state is ACK_UPDATE
                    The Output state is IDLE
BSC externalFunction() at 00:00:02:075, value: -5
                    I received UPDATE_ACK ext message -5   at 00:00:02:075
MSC outputFunction() at 00:00:02:075
MSC internalFunction()
                    The input state is IDLE
                    Call passivate()
                    The Output state is IDLE
BSC outputFunction() at 00:00:02:085
                    I sent update ack at 00:00:02:085
BSC internalFunction()
                    The input state is ACK
                    The Output state is IDLE
BTS externalFunction() at 00:00:02:085, value: -5
                    I received UPDATE_ACK ext message -5   at 00:00:02:085
BSC outputFunction() at 00:00:02:085
BSC internalFunction()
                    The input state is IDLE
                    Call passivate()
                    The Output state is IDLE
BTS outputFunction() at 00:00:02:095
                    I sent update ack at 00:00:02:095
BTS internalFunction()
                    The input state is ACK
                    The Output state is IDLE
MS externalFunction() at 00:00:02:095, value: -5
                    I received FORCE_UPDATE_ACK ext message -5   at 00:00:02:095
BTS outputFunction() at 00:00:02:095
BTS internalFunction()
                    The input state is IDLE
                    Call passivate()
                    The Output state is IDLE
MS outputFunction() at 00:00:02:105
MS internalFunction()
                    The input state is IDLE
                    Call passivate()
                    The Output state is IDLE
Simulation ended!
```

The results shown above is consistent with expectation which could prove the correctness of this entire model system which simulates the location update procedure in GSM mobile network.