



# ENSC 835: OPNET Tutorial

---

Modupe Omueti and Renju Narayanan

Communication Networks Laboratory

<http://www.ensc.sfu.ca/research/cnl>

School of Engineering Science

Simon Fraser University



A decorative graphic on the left side of the slide, featuring overlapping yellow, red, and blue squares with a black crosshair.

# Roadmap

---

- OPNET Modeler
- Settings
- Creating projects
- Creating links
- Node models
- Packet format
- ICI format
- Process model
- Kernel procedures
- Compiling and debugging
- Collecting results



# OPNET modeler

---

- Editors:
  - Project Editor
  - Node Editor
  - Process Editor
  - Link Editor
  - Packet Editor



# Settings

---

- Model directories
- Edit-> Preferences:
  - `bind_shobj_prog`: `bind_so_gcc`
  - `bind_static_prog`: `bind_gcc`
  - `comp_prog`: `comp_gcc`
  - `repositories`: `()`



# Creating projects

---

- Network models: scenarios
- Choosing the size of the network
  - world
  - campus
  - office
  - logical
- Nodes in the network
- Creating object palette
- Trajectories
- Managing scenarios



# Creating links

- Links
  - create links using link editor
  - example: `gprs_llc_link`
- Type of link:
  - point-to-point:
    - simplex – `ptsimp`
    - duplex- `ptdup`
  - bus
- Packet formats supported
- Transmission delay model (`txdel`):
  - point-to-point link: `dpt_txdel`
  - bus: `dbu_txdel`
- Propagation model
- Error model



# Node models

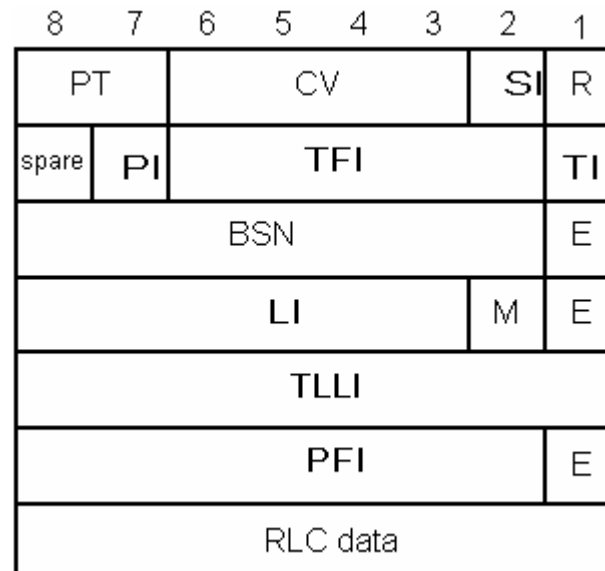
---

- Create your own: e.g., GPRS
- Modify an existing model: e.g., MTCP
- Various modules:
  - processors
  - queues: active, passive
    - first-in-first-out
    - priority
    - last-in-first-out
  - transmitters, receivers, antenna
  - packet stream
  - statistic wires



# Packet format

- Packet editor
- KP: op\_pk\_create\_fmt()
- Fields: length could be zero
- Set and unset fields inside code



Uplink RLC data block





# Process model

---

- States
- Forced and unforced states
- Transitions
- Enter and exit executives
- State variables
- Temporary variables
- Header block
- Function block
- Include files (.h)



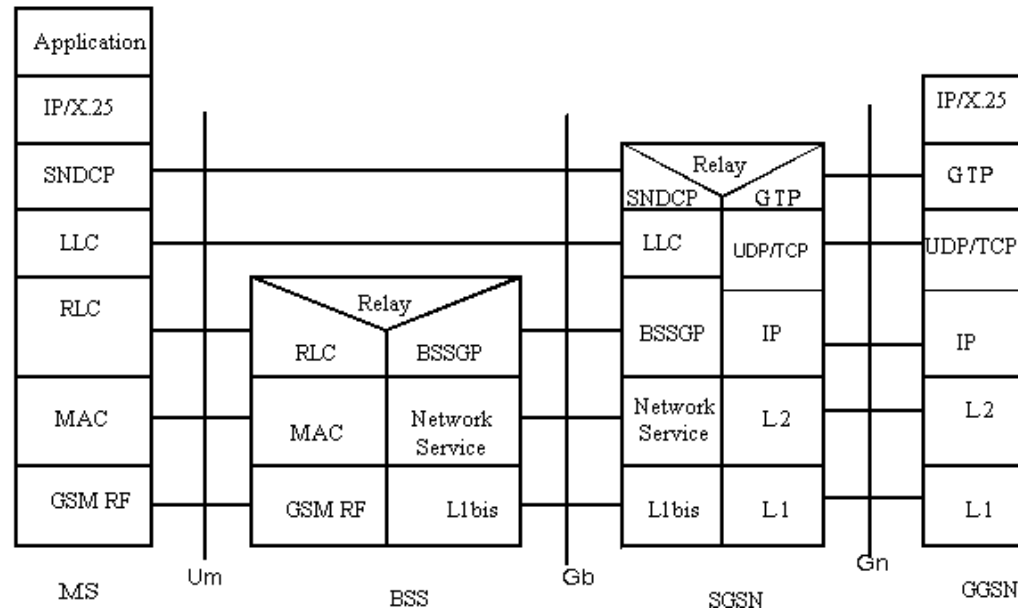
# General Packet Radio Service (GPRS)

---

- Packet switched service
- GPRS uses a combination of Time Division Multiple Access (TDMA) and Frequency Division Multiple Access (FDMA) schemes
- Direction of data transfer:
  - Mobile Station (MS) to Base Station Subsystem (BSS): uplink
  - BSS to MS: downlink



# GPRS: transmission plane



SNDCP: Sub Network Dependent Convergence Protocol

LLC: Logical Link Control layer

RLC: Radio Link Control

MAC: Medium Access Control

BSSGP: Base Station Subsystem GPRS Protocol

GTP: GPRS Tunneling Protocol



# GPRS: RLC/MAC layer

---

- Radio Link Control layer:
  - segments and reassembles LLC PDUs into RLC/MAC blocks
  - acknowledged operation
  - unacknowledged operation
- Medium Access Control layer:
  - controls the allocation of channels and timeslots
  - multiplexes data and control signals
  - provides contention resolution

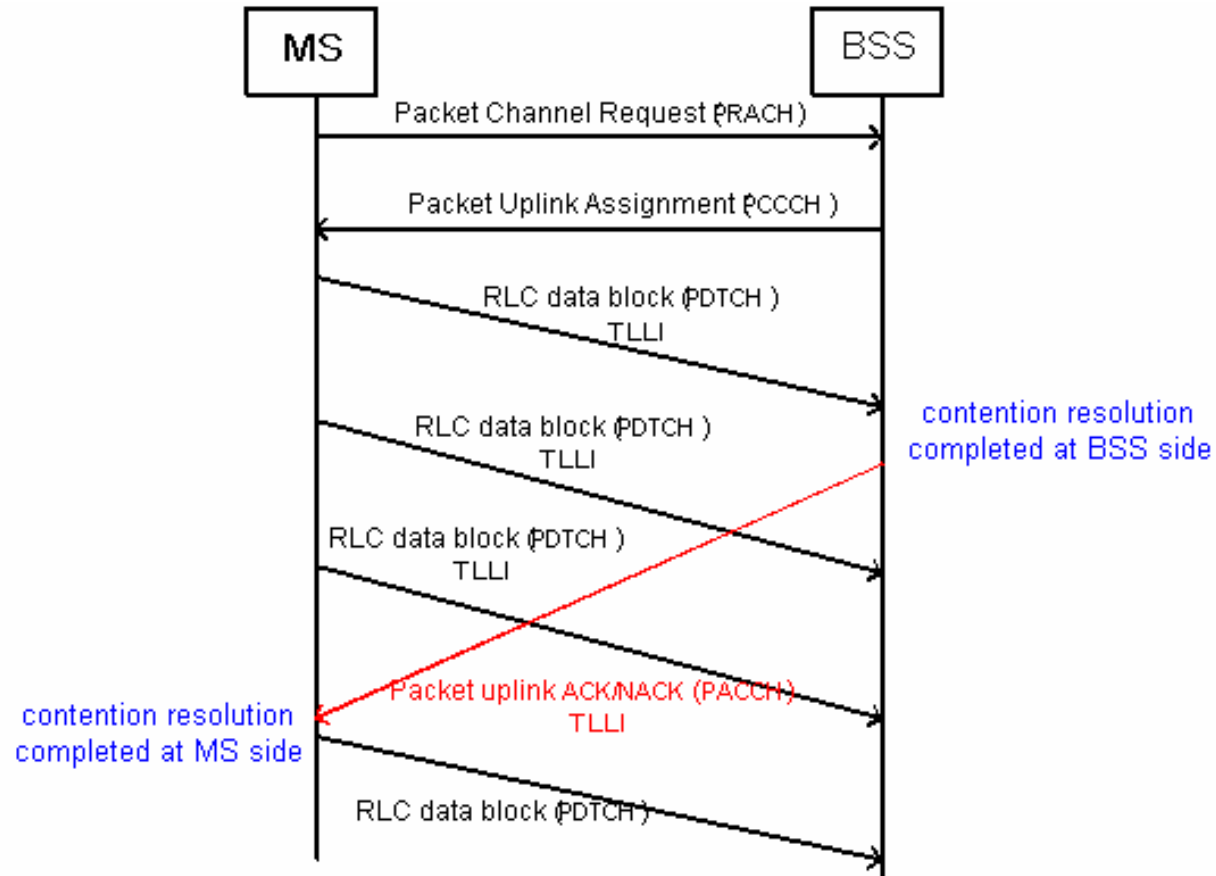


# RLC/MAC parameters

---

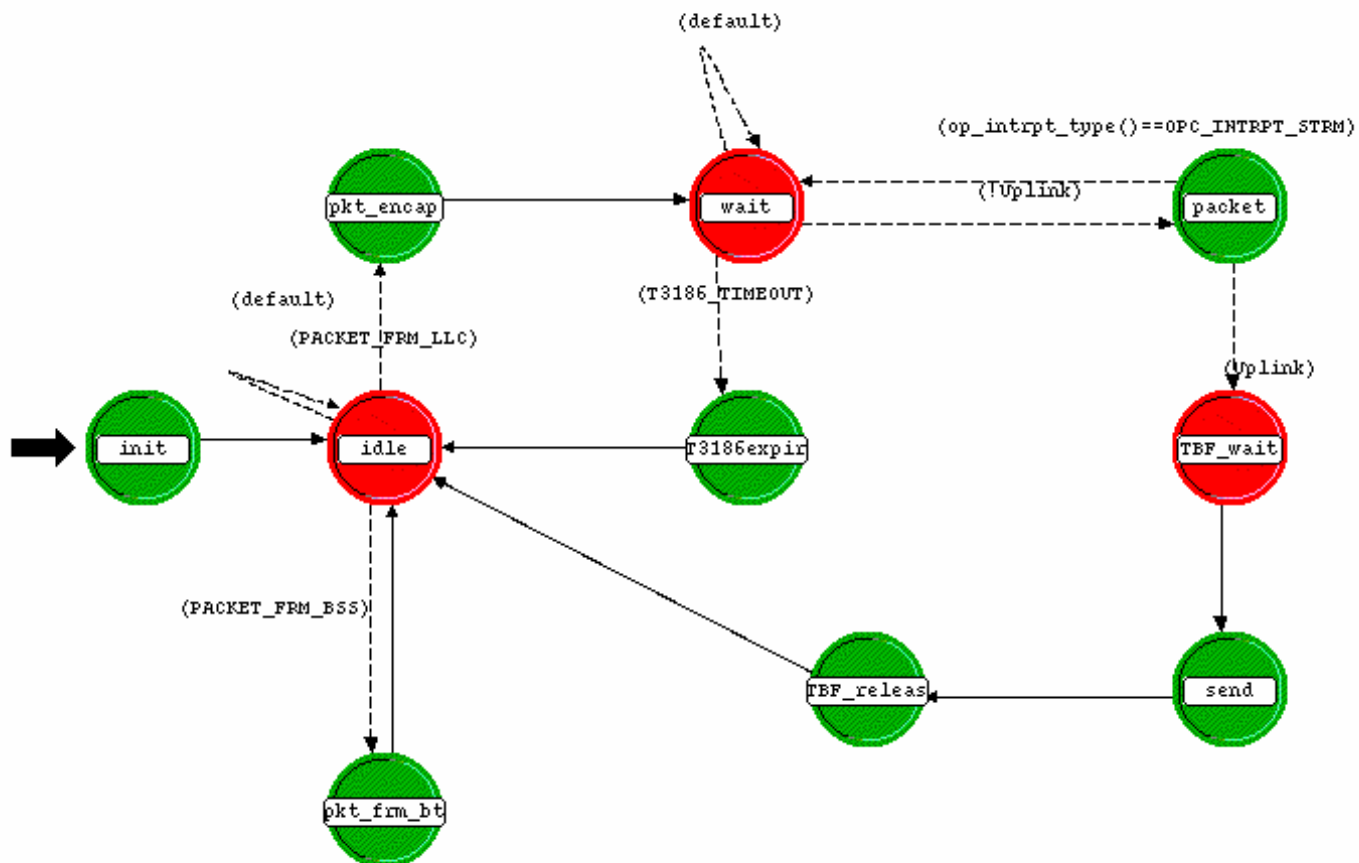
- Temporary Block Flow (TBF): physical connection used by two radio resource entities to support unidirectional data transfer on physical channels
  - downlink and uplink TBF
  - temporary
  - maintained for the duration of data transfer only
- Network assigns a Temporary Flow Identity (TFI) to each TBF
  - TFI is unique among TBFs in the same direction

# One phase access and contention resolution



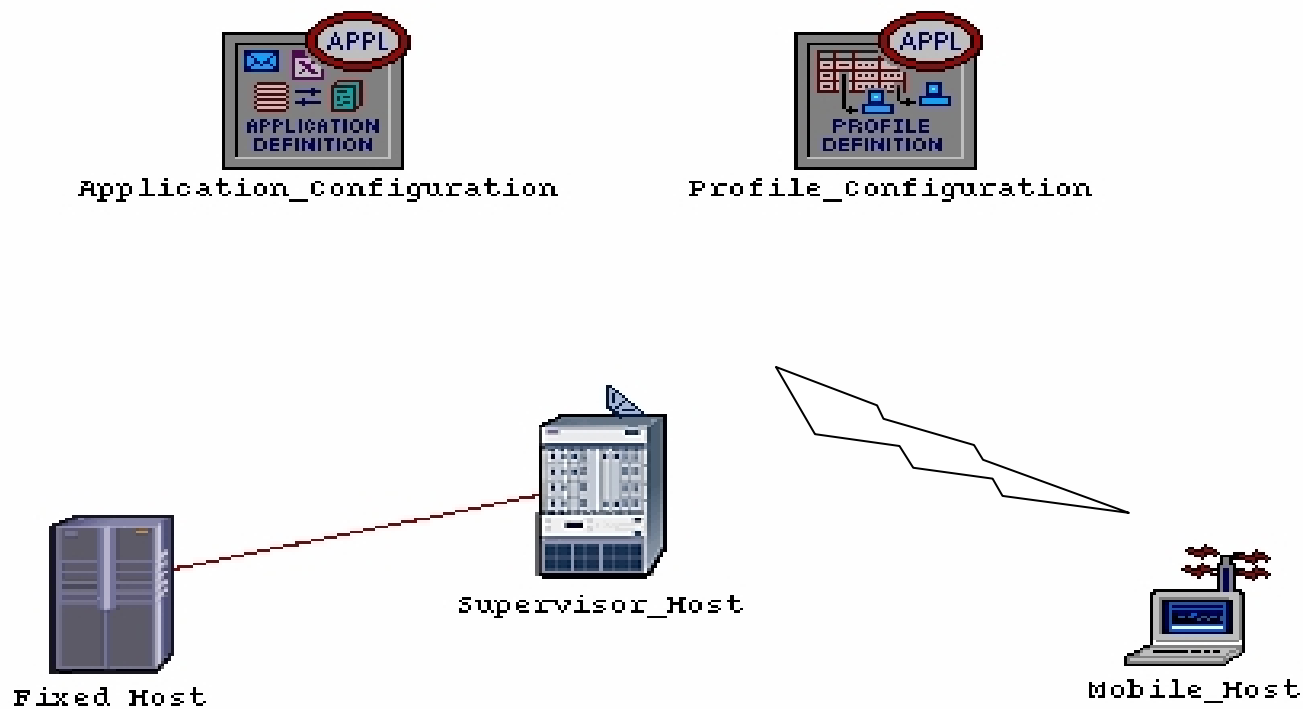


# Process model: RLC/MAC (MS)





# M-TCP







# M-TCP design considerations

---

- Dynamic change of bandwidth in cells
- Frequent periods of disconnection
- Scarce power resources



# M-TCP protocol characteristics

---

- Maintain end-to-end semantics of TCP
- Adapt to dynamic bandwidth changes
- Deal with disconnections
- Ensure efficient handoffs



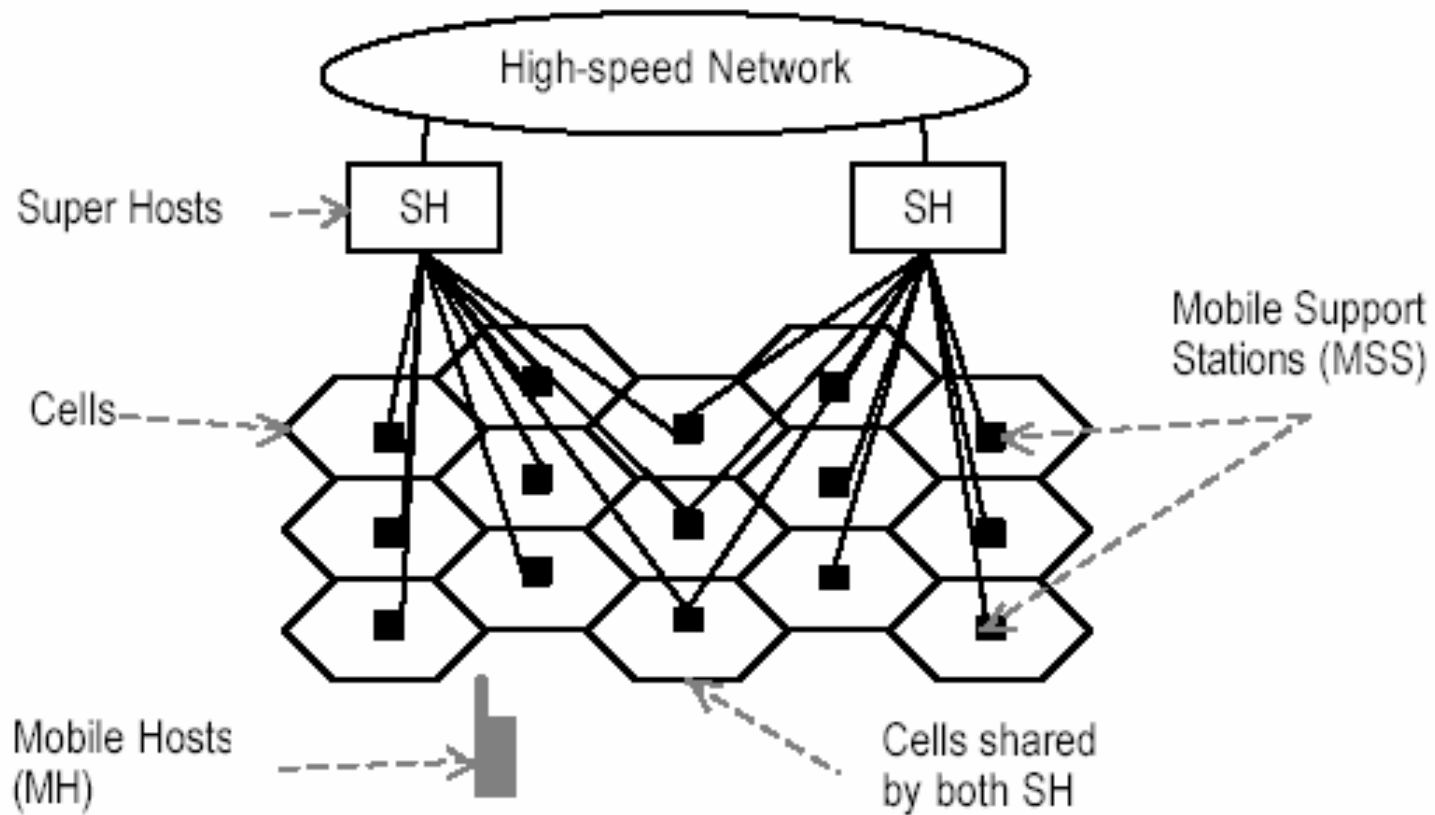
# M-TCP mode of operation

---

- Freezes all timers when disconnections occur
- Monitors wireless link connectivity
- Puts sender (fixed host) into persist mode
- Sets receiver's (mobile host) *cwnd* to zero
- Resumes with previous sender's *cwnd*

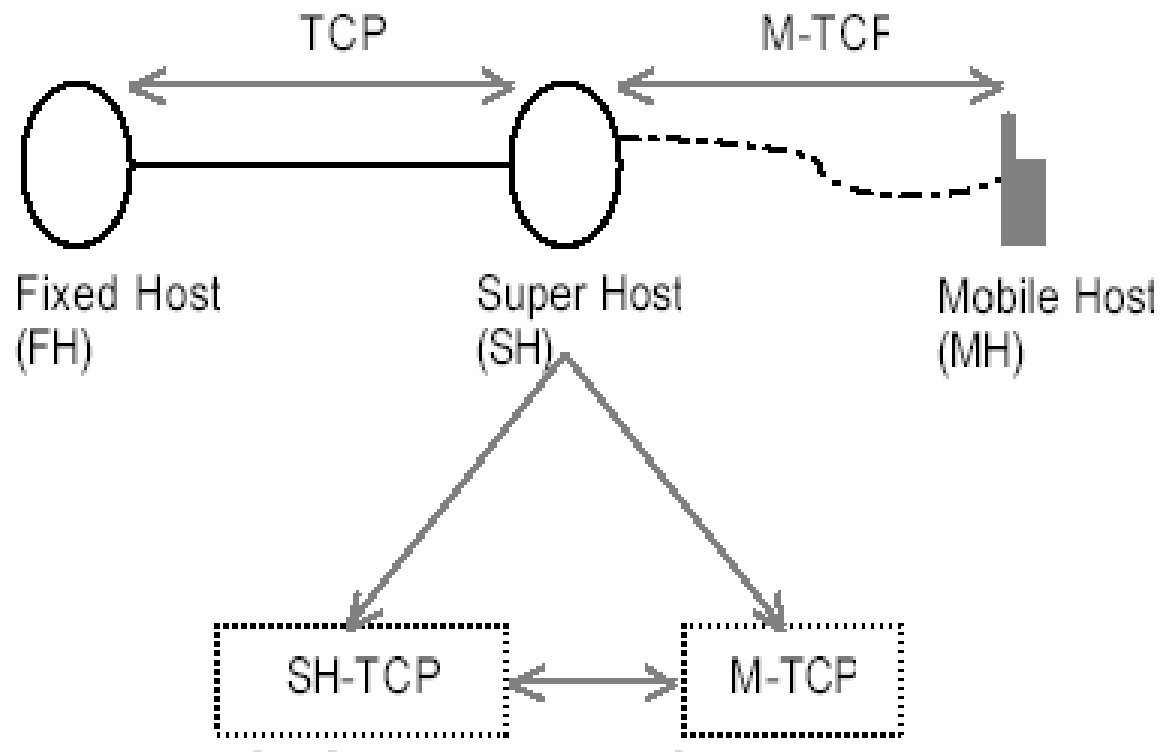


# Mobile network architecture for M-TCP





# M-TCP split connection





# M-TCP parameters

---

- File size
- Inter-request time and distribution
- Sender Maximum Segment Size (SMSS)
- Note:
  - If the file size is small and auto assigned is selected for the SMSS, then the congestion window would be constant.
  - The explanation for this behavior is that the file is downloaded in so short a time that the congestion window did not need to increase in size.

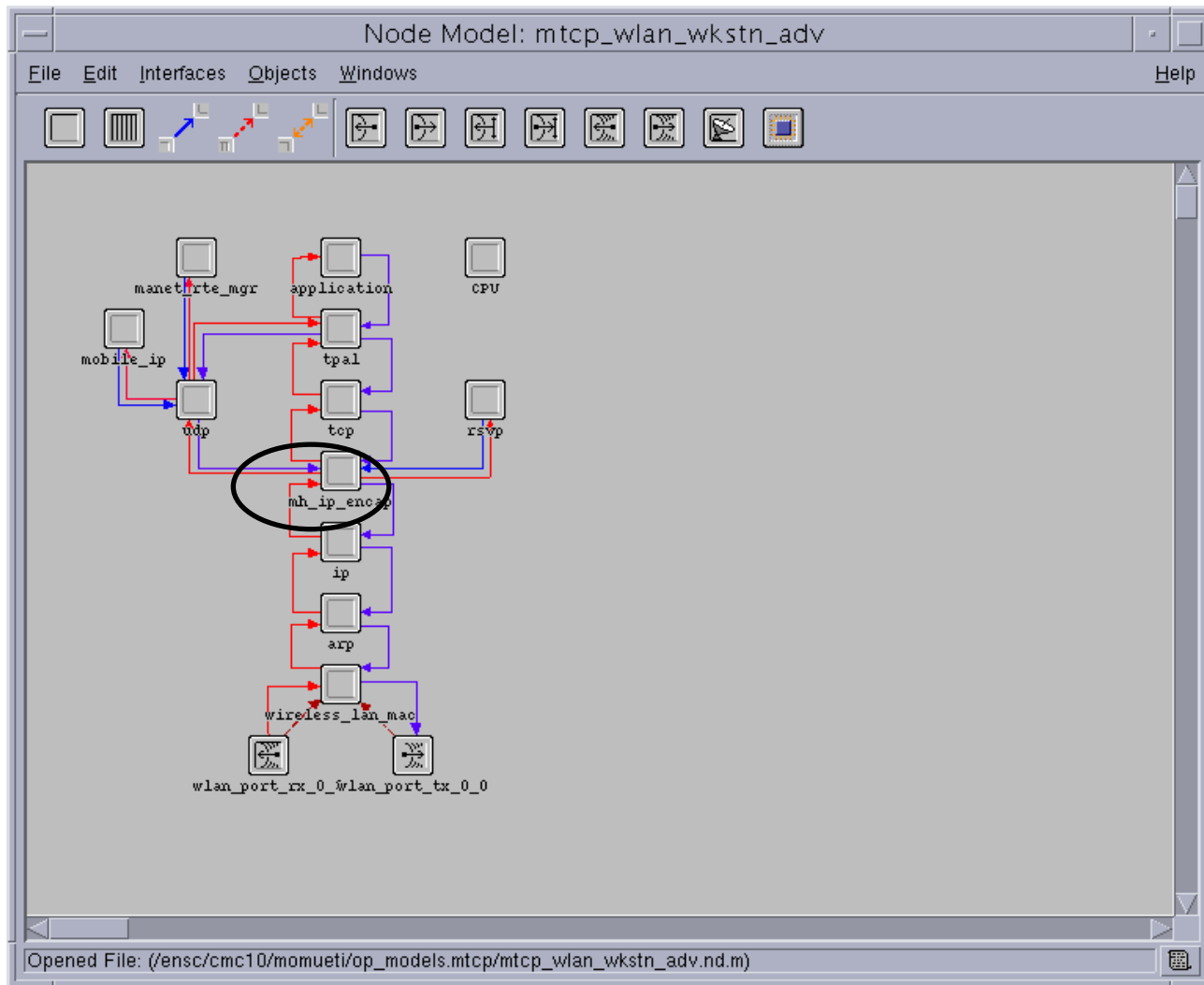


# Application definition

| Attribute                    | Value                 |
|------------------------------|-----------------------|
| Command Mix (Get/Total)      | 100%                  |
| Inter-Request Time (seconds) | constant (180)        |
| File Size (bytes)            | constant (1600000...) |
| Symbolic Server Name         | FTP Server            |
| Type of Service              | Best Effort (0)       |
| RSVP Parameters              | None                  |
| Back-End Custom Application  | Not Used              |



# Mobile host node model







# Modified ip\_encap process attributes

| Attribute       | Value               |
|-----------------|---------------------|
| name            | mh_ip_encap         |
| process model   | mtcp_mh_ip_encap_v4 |
| icon name       | processor           |
| BrokenEnable    | promoted            |
| BrokenEndTime   | 300                 |
| BrokenStartTime | 270                 |
| CycleTime       | 300                 |

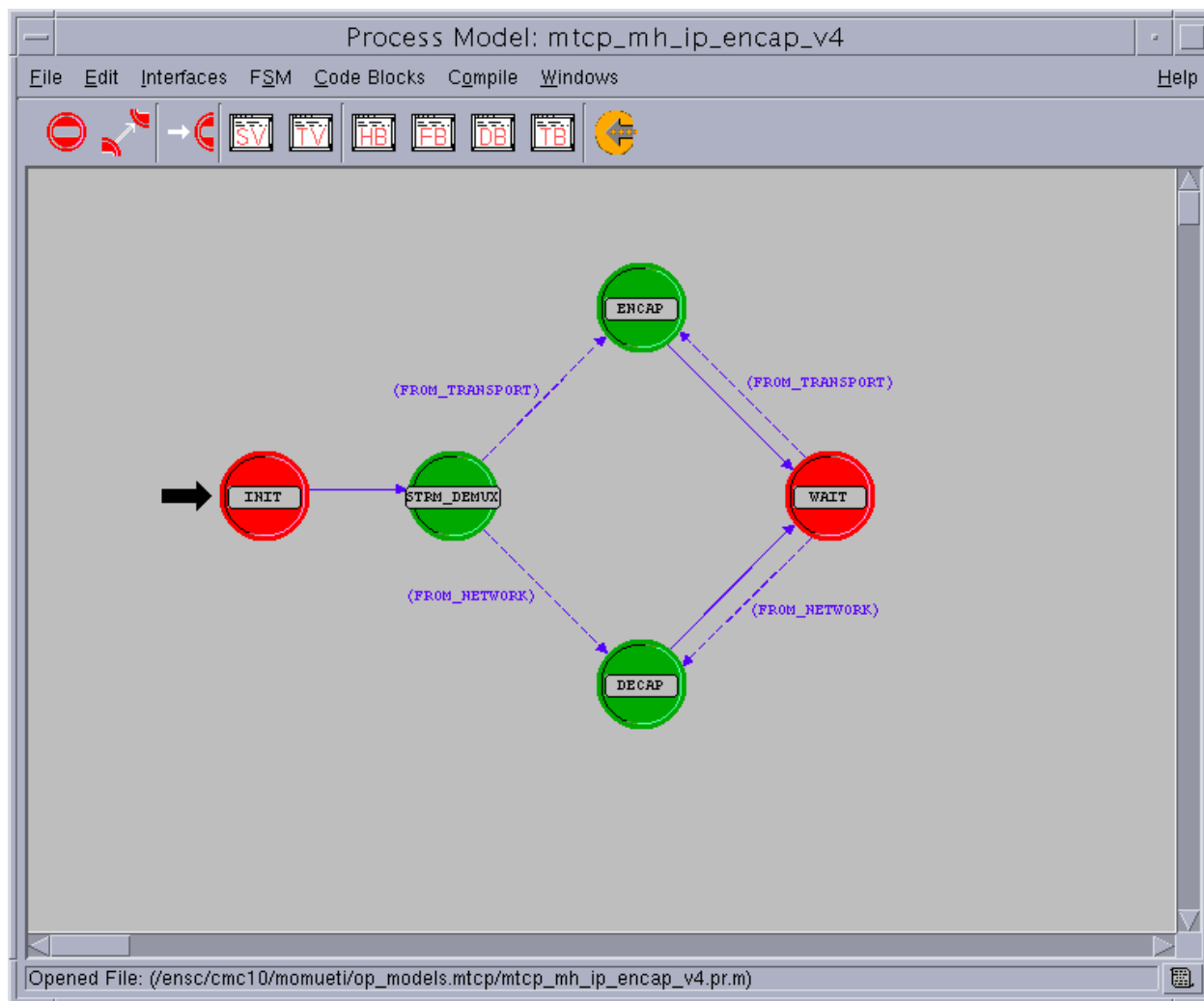
Extended Attrs.

Apply changes to selected objects

Find Next OK Cancel

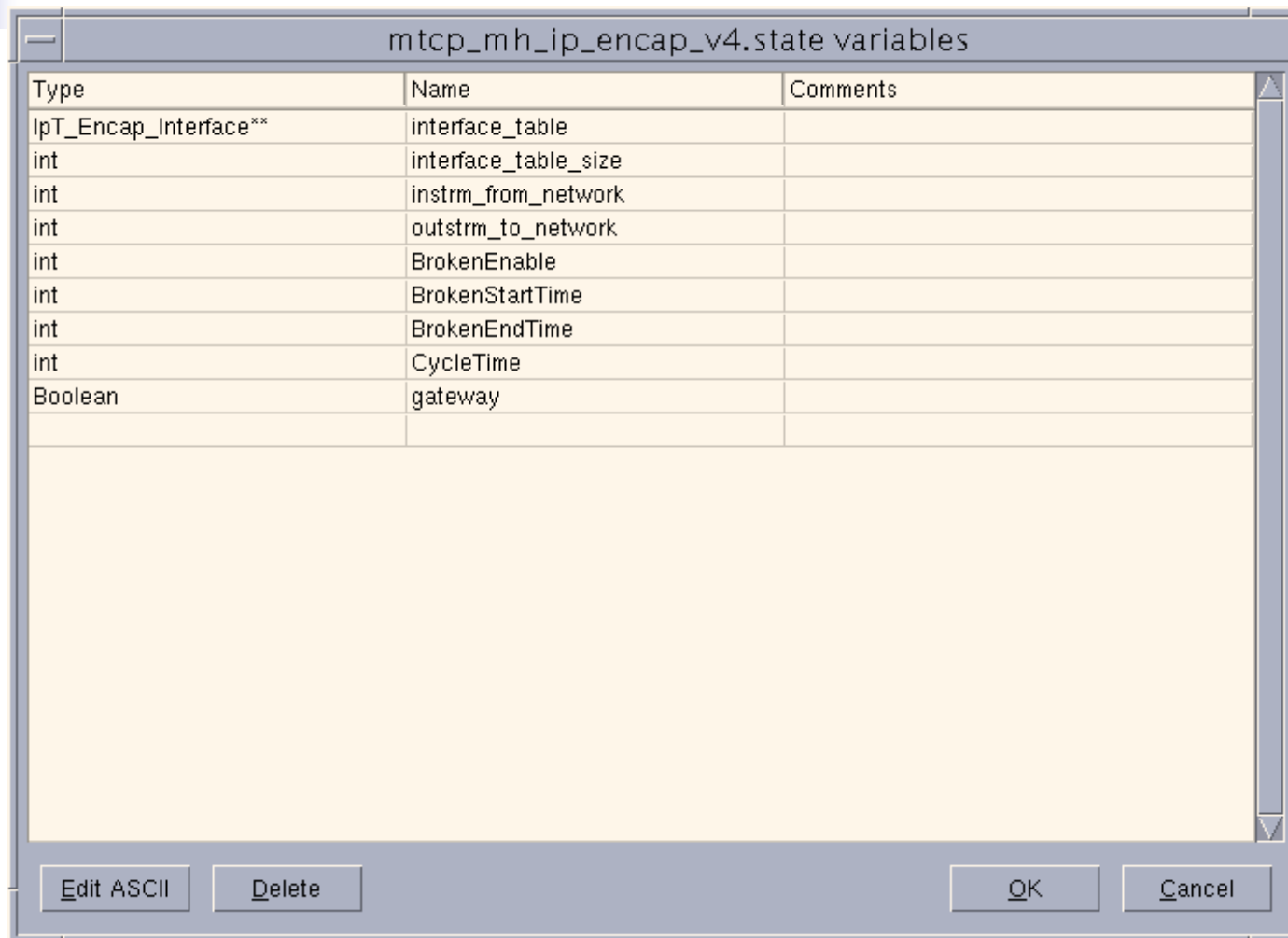


# Modified ip\_encap process model





# Modified ip\_encap state variables

A screenshot of a NetSim window titled 'mtcp\_mh\_ip\_encap\_v4.state variables'. The window contains a table with three columns: 'Type', 'Name', and 'Comments'. The table lists several state variables, including 'interface\_table' (Type: IpT\_Encap\_Interface\*\*), 'interface\_table\_size' (Type: int), 'instrm\_from\_network' (Type: int), 'outstrm\_to\_network' (Type: int), 'BrokenEnable' (Type: int), 'BrokenStartTime' (Type: int), 'BrokenEndTime' (Type: int), 'CycleTime' (Type: int), and 'gateway' (Type: Boolean). At the bottom of the window, there are four buttons: 'Edit ASCII', 'Delete', 'OK', and 'Cancel'.

| Type                  | Name                 | Comments |
|-----------------------|----------------------|----------|
| IpT_Encap_Interface** | interface_table      |          |
| int                   | interface_table_size |          |
| int                   | instrm_from_network  |          |
| int                   | outstrm_to_network   |          |
| int                   | BrokenEnable         |          |
| int                   | BrokenStartTime      |          |
| int                   | BrokenEndTime        |          |
| int                   | CycleTime            |          |
| Boolean               | gateway              |          |



# Modified ip\_encap function block

```
mtcp_mh_ip_encap_v4.function block
File Edit Options
[Icons]
1  /******
2  /* Addition to implement MTCP */
3
4  long      cycle;
5
6  int isBroken ()
7  {
8      double t = op_sim_time();
9      long tt = t;
10     cycle = tt%CycleTime;
11
12     if ((cycle>BrokenStartTime) && (cycle<BrokenEndTime) && (BrokenEnable == 1))
13         return 1;
14 }
15
16 /* Addition ends */
17 /******
18
19 static int
20 ip_encap_proto_get (int strm_num, int num_iface, IpT_Encap_Interface** interface_table_
21 {
22     int      i;
23
24     /** This function returns the protocol type corresponding to an input stream index.
25     FIN (ip_encap_proto_get (strm_num, num_iface, interface_table_handle))
26
```



# Modified ip\_encap init state

```
mtcp_mh_ip_encap_v4 : INIT : Enter Execs
File Edit Options
[Icons]
9  /******
10 /* Addition to implement the cyclic disconnection mechanism of MTCP */
11 /* Obtain all the attributes of the mtcp_mh_ip_encap process model. */
12 /* which are setup parameters of the simulation scenarios. */
13 /* BrokenEnable -- determines whether or not a broken link is */
14 /* allowed during simulation. */
15 /* CycleTime -- determines the time duration of a cycle in which */
16 /* a broken link is enabled */
17 /* BrokenStartTime -- indicates time to begin broken link status */
18 /* BrokenEndTime -- indicates time to end broken link status */
19 /******
20
21 if (op_ima_obj_attr_get( op_id_self(), "BrokenEnable", & BrokenEnable) != OPC_COMPCODE_SUCCESS)
22 printf ("Unable to get this attribute BrokenEnable\n");
23 if (op_ima_obj_attr_get( op_id_self(), "BrokenStartTime", & BrokenStartTime) != OPC_COMPCODE_SUCCESS)
24 printf ("Unable to get this attribute BrokenStartTime\n");
25 if (op_ima_obj_attr_get( op_id_self(), "BrokenEndTime", & BrokenEndTime) != OPC_COMPCODE_SUCCESS)
26 printf ("Unable to get this attribute BrokenEndTime\n");
27 if (op_ima_obj_attr_get( op_id_self(), "CycleTime", & CycleTime) != OPC_COMPCODE_SUCCESS)
28 printf ("Unable to get this attribute CycleTime\n");
29 printf ("LinkBroken=%d, StartTime=%d, EndTime=%d, Cycle=%d\n", BrokenEnable, BrokenStartTime, BrokenEndTime, CycleTime);
30
31 /****** Addition Ends *****/
32
33
34 /** Register using OMS Process Registry. **/
35
36 /* Obtain the necessary objids. */
37 own_objid = op_id_self ();
```

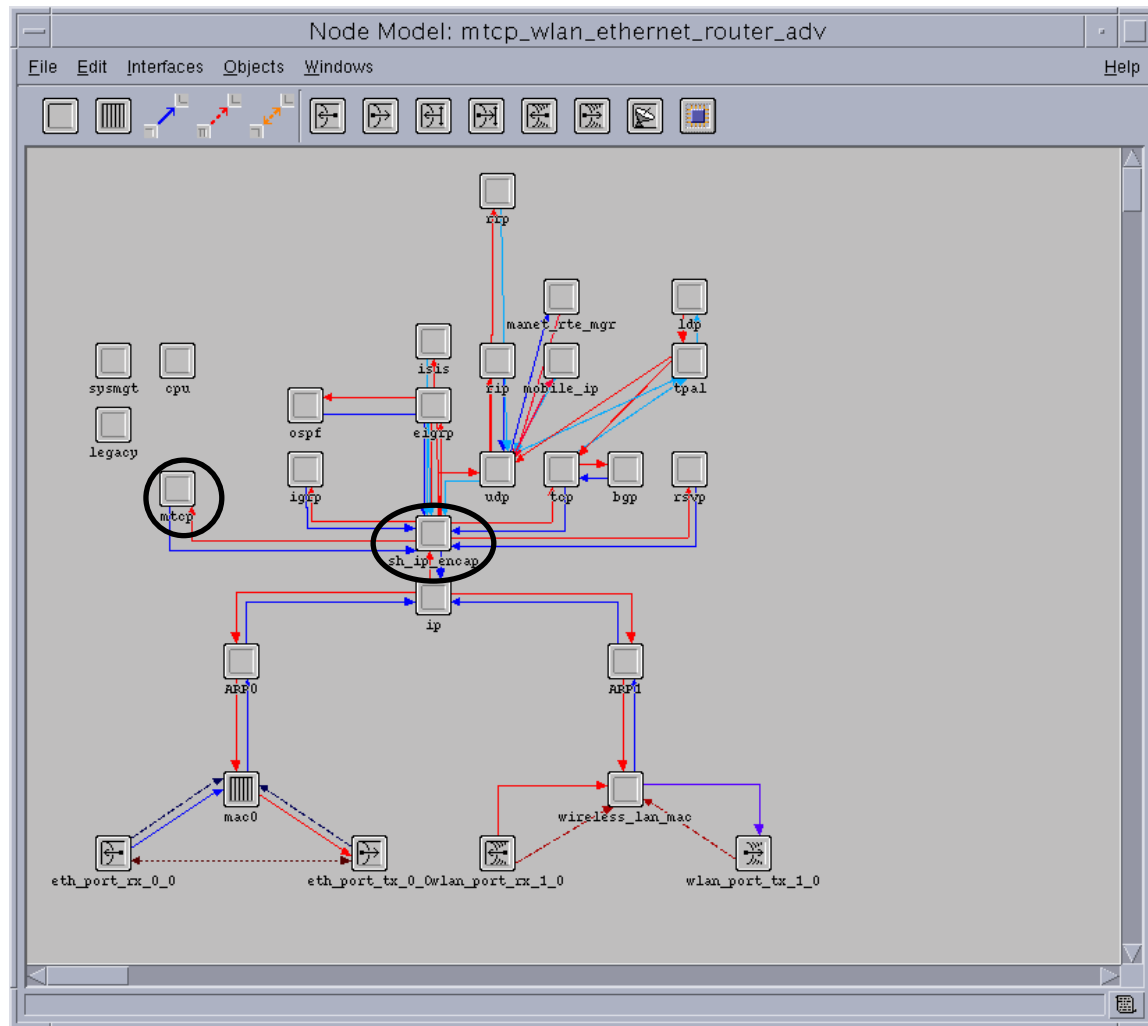


# Modified ip\_encap decap state

```
mtcp_mh_ip_encap_v4 : DECAP : Enter Execs
File Edit Options
op_ici_attr_set (transp_iciptr, "iface_load",      iface_load);
op_ici_attr_set (transp_iciptr, "iface_speed",    iface_speed);
op_ici_attr_set (transp_iciptr, "iface_reliability", iface_reliability);
}
241
242 /******
243 /* Addition to implement the cyclic disconnection mechanism of MTCP */
244
245 if ( isBroken())
246 {
247     printf ("***** link is broken now.*****");
248
249     /* Discard the IP packet. */
250     op_pk_destroy (ip_pkptr);
251 }
252 else
253
254 /****** Addition ends *****/
255 {
256     /* Install the ICI and send the packet to the */
257     /* higher layer. */
258     op_ici_install (transp_iciptr);
259     op_pk_send (pkptr, output_strm);
260 }
261
262 }
263
264
```

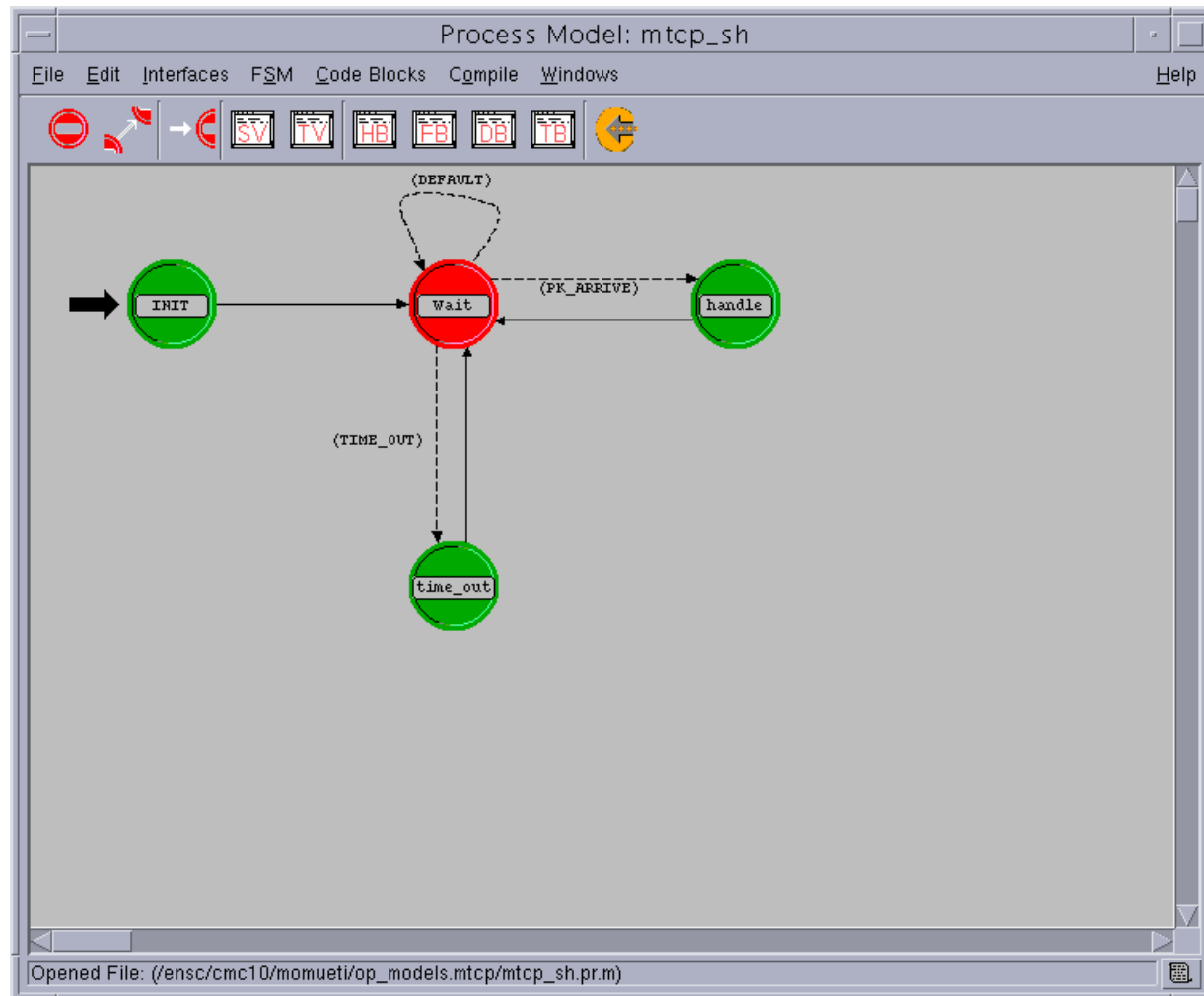


# Supervisor host node model





# M-TCP process model







# M-TCP process header block

```
mtcp_sh.header block
File Edit Options
1 /* Header Block */
2 #include <opnet.h>
3
4 #include <ip_addr_v4.h>
5 #include <oms_dt.h>
6 #include <tcp_api_v3.h>
7 #include <tcp_v3.h>
8 #include <tcp_support.h>
9 #include <oms_pr.h>
10 #include <oms_qa.h>
11 #include <oms_tan.h>
12 #include <tcp_seq_sup.h>
13 #include <ip_notif_log_support.h>
14 #include <ip_rte_v4.h>
15 #include <ip_higher_layer_proto_req_sup.h>
16 #include <ip_qos_support.h>
17
18 static Ici * iciptr;
19 static Packet * pkptr;
20
21 /* Define transition macros */
22
23 #define PK_ARRIVE ( op_intrpt_type() == OPC_INTRPT_STRM )
24 #define TIME_OUT ( (op_intrpt_type() == OPC_INTRPT_REMOTE ) || (op_intrpt_type() ==
25 OPC_INTRPT_SELF ) )
26 #define DEFAULT ( (op_intrpt_type() != OPC_INTRPT_STRM) && (op_intrpt_type() !=
27 OPC_INTRPT_REMOTE ) && (op_intrpt_type() != OPC_INTRPT_SELF))
28
29 #define FROM_ENCAP_STRM 0
30 #define TO_ENCAP_STRM 0
31
32 #define FIXED_HOST_ADDR 0xc0000009
33 #define MOBILE_HOST_ADDR 0xc0000101
34
35 static int isInit =0;
36
37 static int fh_data_len;
38
39 static unsigned fixed_host_port ;
40 static unsigned mobile_host_port;
41
42 unsigned fh_last_seq_num;
43 unsigned fh_last_ack_num;
44
45 // the ack from Mobile host, seen by router, which is
46 // ack seen by Fixed host plus 1
47 unsigned mh_last_ack_num;
48 unsigned mh_last_seq_num;
49
50 // shared variables for incoming ip pkt
51 static IpT_Address org_addr;
52 static IpT_Address dest_addr;
53 static int type_of_service;
54 static int protocol_type ;
```



# M-TCP process function block

```

mtcp_sh.function block
File Edit Options
212 if (op_pk_nfd_access (original_pkt, "fields", &tcp_seg_fd_ptr) == OPC_COMPCODE_FAILURE)
213 {
214 printf ("modify MH: Unable to get data from TCP Segment to modify top pkt.\n");
215 }
216 tcp_seg_fd_ptr->ack_num = ack_num;
217 }
218
219
220 ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
221 //
222 // Generate an ACK packet with rcv_wnd=0, this packet is used to force
223 // the sender into persist mode
224 //
225 ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
226 static Packet * generate_persist_ACK( )
227 {
228 Packet * seg_ptr;
229 TopT_Seg_Fields * tcp_seg_fd_ptr;
230
231 // duplicate the sample ACK packet
232 seg_ptr = op_pk_copy ( sample_MH_pkt );
233
234 // get the tcp control fields
235 if (op_pk_nfd_access (seg_ptr, "fields", &tcp_seg_fd_ptr) == OPC_COMPCODE_FAILURE)
236 {
237 printf ("Unable to get data from TCP Segment to get persist top pkt.\n");
238 }
239
240 // set the tcp control fields and make sure rcv_wnd=0

```



# M-TCP process function block

```
mtcp_sh.function block
File Edit Options
455 // check to see if the link is broken
456 if ( isBroken() )
457 {
458 {
459 // first time in persist mode
460 if ( transMode == 0 )
461 {
462 // this is the flag indicating the transition into persist mode
463 transMode = 1;
464
465 // which cycle do we start the persist mode, cycle refers to brokenBeginCycle and
466 transBeginCycle=cycle;
467
468 // for debug
469 printf( stderr, "send ACK for persist mode at %lf*****\n", op_sim_time() );
470 printf("send ACK for persist mode");ispatch,indicating the */
471 e packet
472 Packet* persist_ACK = generate_persist_ACK();
473
474 // send persist mode packet
475 op_pk_send( myEncap( persist_ACK), TO_ENCAP_STRM );
476 }
477 else printf("pk not sent for persist mode" );
478 }
479
480 else
481 {
482 // not in persist mode
483 transMode=0;
484 // reduce the ACK number by 1
485 modify_MH_pkt( ip_pkptr, mh_last_ack_num-1 );
486 // send the modified ACK packet to FH
487 op_pk_send(ip_pkptr, TO_ENCAP_STRM );
488 }
}
Line: 1
```



# Some kernel procedures (KPs) used

---

- Packet processing:
  - Op\_pk\_get()
  - Op\_pk\_nfd\_set()
  - Op\_pk\_nfd\_get()
  - Op\_pk\_send()
- Interrupt processing:
  - Op\_pk\_intrpt\_type()
  - Op\_pk\_intrpt\_strm()
  - Op\_pk\_intrpt\_schedule\_self()



# Some kernel procedures (KPs) used

---

- Segmentation and reassembly
  - Op\_sar\_segbuf\_pk\_insert()
- Queues
  - Op\_subq\_pk\_remove()



# Compiling and debugging

---

- OPNET debugger
- Print statements
- Error file

A decorative graphic on the left side of the slide, featuring overlapping yellow, red, and blue squares with a black crosshair.

# Collecting statistics

---

- Global statistics
- Local statistics

A decorative graphic on the left side of the slide, featuring overlapping yellow, red, and blue squares with a black crosshair.

# Animation

---

- Selecting animation





# Running scenarios

---

- Creating a simulation set

A decorative graphic on the left side of the slide, featuring a vertical black line intersected by a horizontal black line. To the left of the vertical line are three overlapping squares: a yellow one at the top, a red one in the middle, and a blue one at the bottom. The text 'Viewing results and animation' is positioned to the right of this graphic.

# Viewing results and animation

---

- Comparing scenarios
- Playing animation



# Cleaning up

---

- Files that could be deleted to get more space:
  - .ah
  - .ov
  - temporary files
  - backup files
  - Error files



# References

---

- E. Seurre, P. Savelli, and P. Pietri, *GPRS for Mobile Internet*. Norwood, MA: Artech House, 2003 .
- 3rd Generation Partnership Project, TS 04.60 version 7.9.0 General Packet Radio Service (GPRS) Radio Link Control/Medium Access Control (RLC/MAC) layer specification.
- K. Brown and S. Singh, "M-TCP: TCP for mobile cellular networks," *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 5, pp. 19-42, Oct. 1997.
- W. G. Zeng, M. Zhan, Z. Li, and Lj. Trajkovic, ``Performance evaluation of M-TCP over wireless links with periodic disconnections," *OPNETWORK 2003*, Washington, DC, Aug. 2003.
- OPNET documentation V.11.0.A, OPNET Technologies, Inc., Bethesda, MD, 2004.