DEPARTMENT OF COMPUTER SCIENCE
COLLEGE OF SCIENCES
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

A FLEXIBLE SYSTEM FOR SIMULATING AERONAUTICAL TELECOMMUNICATION NETWORK

By
Kurt Maly, Principal Investigator
C. M. Overstreet, Co-Principal Investigator
R. Andey, Co-Principal Investigator

FINAL REPORT

For the period ending December 31, 1998

Prepared for
NASA Langley Research Center
Attn.: Mr. R. Todd Lacks
Grants Officer
Mail Stop 126
Hampton, VA 23681-0001

Under
NASA Research Grant NAG-1-2037, Supplement No.1
Benchmarking the Future Automated Air Traffic Management System
ODURF #182941

December 1998

A Flexible System for Simulating Aeronautical Telecommunication Network

R. Andey, K. Maly, C. M. Overstreet
Computer Science Department
Old Dominion University
Norfolk, VA 23529-0162 USA

{maly, cmo}@cs.odu.edu
(757) 683-4545

ABSTRACT: *At Old Dominion University, we have built ATN (Aeronautical Telecommunication Network) Simulator with NASA being the fund provider. It provides a means to evaluate the impact of modified router scheduling algorithms on the network efficiency, to perform capacity studies on various network topologies and to monitor and study various aspects of ATN through GUI. In this paper we describe briefly about the proposed ATN model and our abstraction of this model. Later we describe our simulator architecture highlighting some of the design specifications, scheduling algorithms and user interface. At the end, we have provided the results of performance studies on this simulator.*

KEY WORDS: *Web-based simulation, scheduling algorithms, network objects, QoS parameters, Capacity studies, and Monitoring system*

I. Introduction

The high level objective of the Aeronautical Telecommunication Network Simulation is to provide a flexible infrastructure to evaluate the impact of modified router scheduling algorithms on the networks efficiency, to perform capacity studies on various network topologies and to monitor the imminent overload situations. The high level requirements can be further divided into the following lower level requirements: A software simulation must be provided to perform this analysis, the user must be able to construct various network models in order to perform capacity studies, the user must be able to construct and run multiple network models simultaneously within the simulation in order to have a comparison base, the user must be able to control which scheduling algorithm to use for each network model in the simulation, the user must be provided with enough feedback to evaluate and compare the impact of the modified scheduling algorithms on the network models and the user must be provided with a monitoring system to report imminent overloaded situations and security threats. The high level architecture of the software simulation is meant satisfying a combination of good engineering qualities like scalability, platform independence, modularity and ease of maintenance.

We have used CSIM18 [6] as our underlying simulation engine in developing our object-oriented simulation server for this model and Java 2D Graph package [7] to display various QoS parameters in the form of strip charts. We have developed a Java Graph Editor package, which provides a flexible interface to create, edit, display and update network scenarios. Mesquite CSIM18 (C++ version) is a process-oriented, general purpose simulation toolkit written with general C language functions. The toolkit allows programmers to create and implement process-oriented, discrete-event simulation models. These models simulate complex systems and offer insight into the system's dynamic behavior. CSIM has cross platform capability and users can develop their models on a PC or Macintosh and then execute them on a RISC workstation for even greater processing power. The Java 2D Graph (Version 2.4) class library is a package of Java classes designed to facilitate plotting data using Java applets/applications.

II. The ATN Primer

The Aeronautical Telecommunication Network (ATN) is a world wide data network with an intention to provide data communications connectivity among mobile platforms, airlines and other companies that provide services such as Aeronautical Operational Control (AOC), Aeronautical Administrative Communications (AAC) and Aeronautical Passenger Communications (APC), and government authorities that provide Air Traffic Control (ATC) and Flight Information Services (FIS). This network is being designed as a collection of dissimilar transmission networks and interconnecting computers that will allow it to operate as a single, cooperative, virtual data network. The goal is to provide full and flexible support for data communications between aviation end-users around the world, both fixed-based and mobile. The ATN is based on standards and guidelines from various standardizing bodies for aviation, including the Airlines Electronic Engineering Committee (AEEC), the Air Transport Association of America (ATA), the International Air Transport Association (IATA), the International Civil Aviation Organization (ICAO), and the RTCA, Inc. These standards and guidelines will in turn be based on the Open Systems Interconnection (OSI) protocols formulated by international standardizing bodies such as the International Organization for Standardization (ISO).

[2] The ATN system consists of interconnected Routing Domains (RDs). The RDs can be classified as Intermediate/Transit RDs (TRDs) and End RDs (ERDs). Each TRD will support the relaying of Network Protocol Data Units (NPDUs) received from one RD to another RD. In other words an ERD will either generate/relay NPDUs or receive/consume NPDUs. Each ATN RD will have a unique identifier, Routing Domain Identifier (RDI). RDs are again classified as Fixed RDs and Mobile RDs. Each State and Organization participating in ATN will operate one or more Fixed RDs in order to interconnect with other RDs. Adjacent States/Organizations can combine their RDs into a single RD. An ATN equipped aircraft will operate one or more Mobile RDs. Both Fixed and Mobile RDS will come under ERD category. The Ground ATN Internet consists of one or more ATN Island RDCs (Routing domain Confederations), That comprises of one or more ATN RDs. An example ATN Island RDC topology is presented in Fig.1.

ATN sub-network is any fixed or mobile data communications network conforms to the following requirements: Sub-networks will transfer data in a byte and code independent manner, provide QoS indication, provide a unique identity to each sub-network, route NPDUs between specified source and destination Sub-network Point of Attachment (SNPA) and support a minimum of 1100 bytes sub-network service data unit (SNSDU). Apart from these requirements, Mobile sub-networks will also conform to the following requirements: mobile sub-networks will provide a sub-network access protocol mechanism for invocation of sub-network priority (QoS), provide QoS parameters include transit delay for different classes, security, cost and error probability, provide a way to allocate resources on a per user or per connection basis, provide a mechanism for detection of change in media connectivity and for the conveyance of this info to connected ATN routers and provide segmentation/reassembly to allow the conveyance of large SNSDU greater than the sub-network's internal packet size between SNPAs.

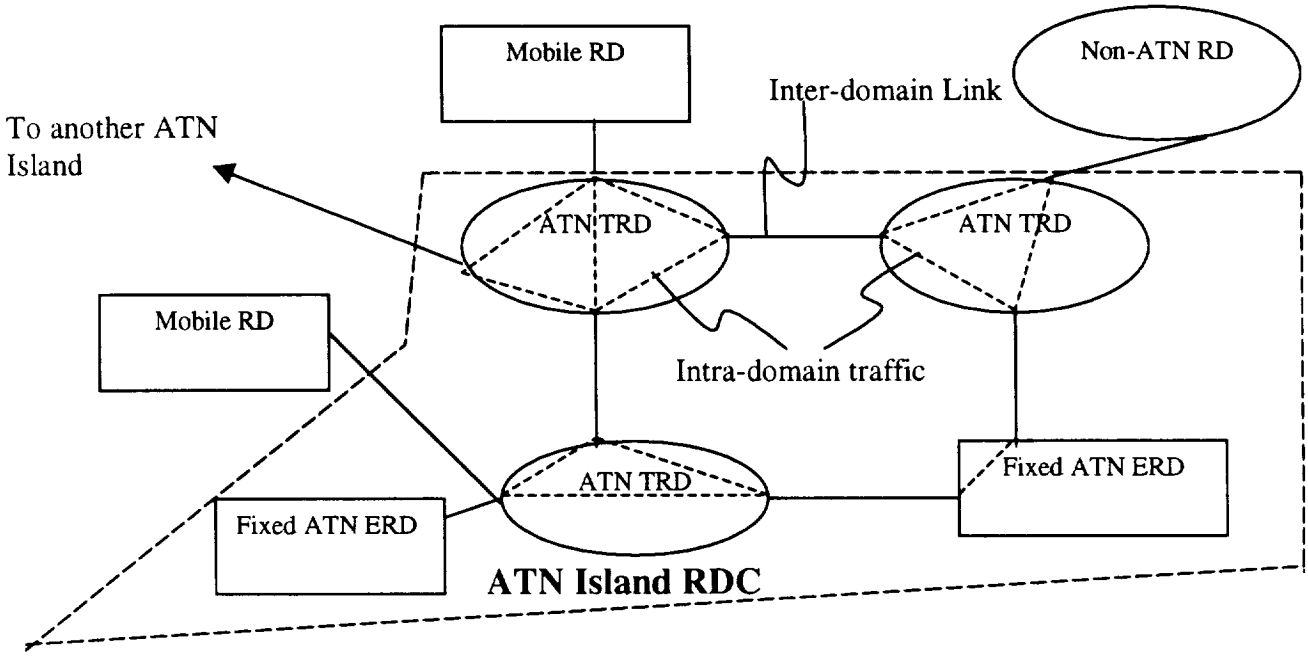

Figure 1. ATN Island Routing Domain Confederation Structure

ATN security will protect DL applications from internal and external threats ensuring QoS and routing policy requirements. Applications have to develop their own security measures. But ATN does ensure QoS and other routing requirements. It uses a Security Label in the header of each CLNP PDU to convey information identifying the 'traffic type' of the data and application's routing policy or QoS. Different traffic types available are ATN Operational Communications, ATN Administrative Communications, General Communications and ATN System Management Communications. ATN Operational Communications traffic will have Air Traffic Service Communications (ATSC) and AOC categories. In the network layer type 2 authentication will enable the routing information base to be protected from attackers that try to modify routing information while in transit, or which attempt to pretend as genuine ATN routers. ATN uses Priority to distinguish the relative importance and urgency of application messages. Any Connection priority won't change during its lifetime. ATN internet entities maintain their queues of outgoing NPDUs in strict priority order, such that a higher priority NPDU in an outgoing queue will always be selected for transmission in preference to a lower priority NPDU. Priority zero is the lowest priority. In connection mode, network layer priority shall mapped onto sub-network priority. More information on ATN system level requirements can be obtained from ATN SARPs, sub-volume 1 [1]. Further details of Internet communications service can be obtained from ATN SARPs, sub-volume 5[2].

III. Simulation Model

Logically our model has been designed in the similar lines of the system explained so far. Any specific requirements missing can be added easily as our model provides that flexibility. Each NPDU is represented by a 'packet' object. The size of a packet is randomly selected between 1KB to several MBs. Packet routing will be done only in pre-specified paths. Each pre-specified routing path is represented by a 'connection' object, which maintains all the routing information form the source of a connection to the destination of a connection. During the simulation run time each 'connection' object will generate packets with interarrival time based on the distribution (e.g. exponential distribution) requested. A 'node' object will present a RD. We did not make any distinction between Mobile RD and Fixed RD. But one can easily extend a node object to present unique properties of different types of RDs. Sub-domains within a RD are represented by 'router' objects. Each priority queue at the input of a router is represented by a 'queue' object, whose size is defined in number of packets. There will be five such priority queues at each router input as we have identified five different types of priority messages in the ATN traffic. A node can have one or more routers within its domain. Different user groups such as ATC, AOC, Sub-systems within a flight and APC are represented by 'host' objects. A node can be attached to any number of hosts. Each Sub-network is represented by a 'link' object. We haven't distinguished between mobile sub-networks and non-mobile sub-networks. Generally a mobile sub-network should have a broadcasting capability. We have assumed all links as point-to-point. Moreover we haven't included segmentation/reassembly mechanism at the links. Table 1. Show different types of links in ATN model. Normally Ethernet is used within a RD. Each ATN Island RDC is represented by a topology object, which can have many nodes, links and connections within its domain. Security concept still has to be explored further in order to introduce into the model. Our model uses Priority to distinguish the relative importance and urgency of application messages. Any Connection priority won't change during its lifetime. Priority zero is the highest priority. Each packet will be queued into corresponding priority queue at the input of routers. Table 2. Show different categories

of traffic exits in ATN. Each packet is selected and delivered to output interface based on the scheduling scheme (Different scheduling schemes available are explained in the next section.) selected.

Link Type	ID	BW (in Mbps)
Mode-S	0	4
SATCOM	1	0.0105
VHF	2	0.0315
ATM	3	150
T1	4	1.5
Ethernet	5	10

Table 1. Different types of links

Traffic Type	Priority
Urgent Communication	0
Flight Safety Message	1
Weather Broadcast	2
Flight Regularity Message	3
Aeronautical Info Service Message	4

Table 2. Different types of ATN traffic

IV. Simulation Architecture

High-Level Architecture

The high level architecture of the software simulation is meant satisfy a combination of good engineering qualities and the student's goals for the learning experience. The engineering qualities are scalability, platform independence, modularity and ease of maintenance. The student's goals are to implement the project using object-oriented methodology, learn C++ along the way and gain experience with web-related technologies. A graphical view of the ATN project's high-level architecture is shown in Fig.2. Java GUI module contains an interface to control the simulation execution, an interface to create, edit, display and update network scenarios and strip chart displays to present feedback information to the user. Simulation Server module contains the entire software infrastructure pertinent to the ATN simulation. Client-server architecture has been implemented between the simulation and the graphical user interface (GUI). This allowed for a loose coupling between the GUI and the simulation, enabling us to build the simulation first, test the simulation in a scalable and incremental manner, and then attached GUI of our choice later in the process. As a result, there is a defined protocol by which the GUI communicates with the simulation server. The Java GUI issues commands over TCP/IP sockets, which we call 'Java SimCmds', are discussed in Appendix A. All the scenario specification and control commands sent from GUI to server are part of 'Java SimCmds'. All simulation statistics and imminent overload situations are reported back to the Java GUI as feedback from the simulation server.

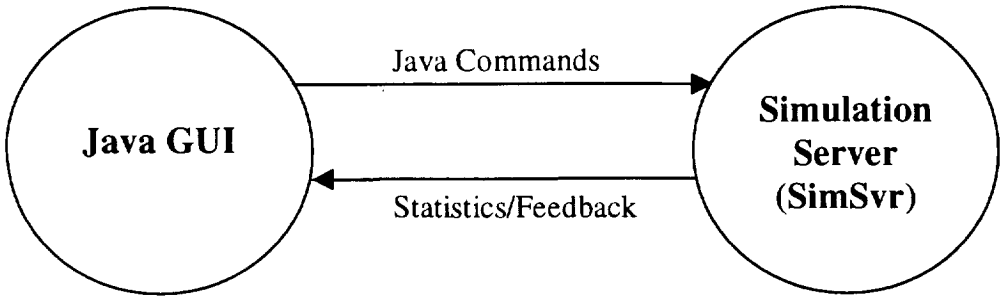

Fig.2. ATN high-level architecture

Simulation Server

The simulation server as an object-oriented server in CSIM18, a C++ based simulation language. CSIM18 is more matured and currently in use by industry. See http://www.mesquite.com for more information on CISM18. We have identified router, connection and packet objects as active resources in the model and realized them using CSIM processes, which are user level threads. Host, link and queue objects are being identified as passive objects. Both host and queue are realized by CSIM facilities whereas CSIM storage objects realize links. In addition to these objects, there is a temporary process, called protocol process, which facilitates the utilization of link storage for a fixed duration (propagation time). A topology object will hold all the network components related to a network scenario. A user can load multiple network scenarios into the simulation server simultaneously and each scenario in turns results in the creation of a new topology object.

The complete life cycle of the simulation is as shown in Fig.3. Each topology object will start all the connection processes and scheduler processes at the routers. A connection process in turn generates packet processes randomly with interarrival time based on the distribution (e.g. exponential distribution). Each packet process lasts for the entire lifetime of a packet. During its lifetime, each packet will visit all the intermediate nodes (routers) apart from the source and destination hosts. When a packet arrives to the input side of a router, enqueue process (packet process itself will become an enqueue process by invoking enqueue method at the router) will add that packet to an appropriate priority queue based on the priority of the packet and wakes up the scheduler process by setting enqueue event and then waits for a dequeue event form the protocol process. Upon receiving an enqueue event, the scheduler process services that packet by placing it on to an appropriate out going link and associates the packet with a protocol process in order for the packet to utilize the link resource during its propagation time. Once the packet propagation is over, protocol process will set dequeue event and wakes up the enqueue process. Then the packet process comes out of enqueue method and proceeds to next hop. This process will continue until it reaches the destination.

Connection Process
Wait until time to start the connection
While (simulation time < connection end time) {
    Generate a Packet Process
    Wait until time to generate another packet
}
Packet Process
If (packet is on first hop) {
    Use the source host resource
}
While (packet is not on last hop) {
    Determine the next router/node resource
    ENQUEUE itself onto that router
    Increment the hop count
}
If (packet is on last hop) {
    Use the destination host resource
    Update the bytes received on the connection
    Calculate delay
    Update bytes received and delay
}
ENQUEUE
Add packet to correct priority queue
If the add was successful {
    Increment number of packets received
    Reserve the input queue facility
    Wait for the time it takes to enqueue
    Set ENQUEUE EVENT // Wake Scheduler
    Release the input queue facility
    Wait until DEQUEUE EVENT is set
}

Scheduler Process
Wait until ENQUEUE EVENT
While not end of simulation {
    Get next packet using scheduling algorithm
    If successful and packet not on last hop {
        Put packet on next available link
        with right src and dest, waiting
        until one is available
    Launch Protocol Process
    Put Scheduler Process to wait until
    next ENQUEUE EVENT
    }
    If no packets are on the queue wait until
        ENQUEUE EVENT
    If packet is on its last hop
        Launch the Protocol Process
        without allocating any link
}
Protocol Process
Wait for time it takes to dequeue
If a link is allocated { // packets not on last hop
    Wait until packet travel time
        // prop. delay
    Release the link resource
}
set DEQUEUE EVENT

Fig.3. The algorithm used for the simulation

Server is completely command driven and controllable from the user interface. There is a dedicated process at the server always listening to a passive TCP socket expecting some command from user interface in the form of simulation commands (SimCmds, refer APPENDIX – B). The simulation commands can be broadly divided into three categories. 1. Commands during a network scenario creation. 2. Commands requesting continuous / instantaneous statistics of any network parameter of interest. 3. Simulation control commands. Scenario creation commands include adding a network component along with its parameters, deleting a network component and setting a scheduling scheme. Continuous statistics requests are the statistics requests made before initiating the simulation run and require updates at regular intervals during simulation run. Whereas instantaneous statistics requests are the statistics requests made during the simulation and require statistics at that particular instant of time. Normally control commands include setting simulation time, which can be alterable at any time during simulation run, warm-up period, which will come in handy to eliminate transient characteristics and simulation speed, which intern will have two components to specify how long the simulation run should proceed before taking a break (break_after_time) and how long should the simulation get suspended at each break point before

resuming the run (break_for_time). Fig.4. Shows different threads/processes invoked at the server once the user issues run simulation command.

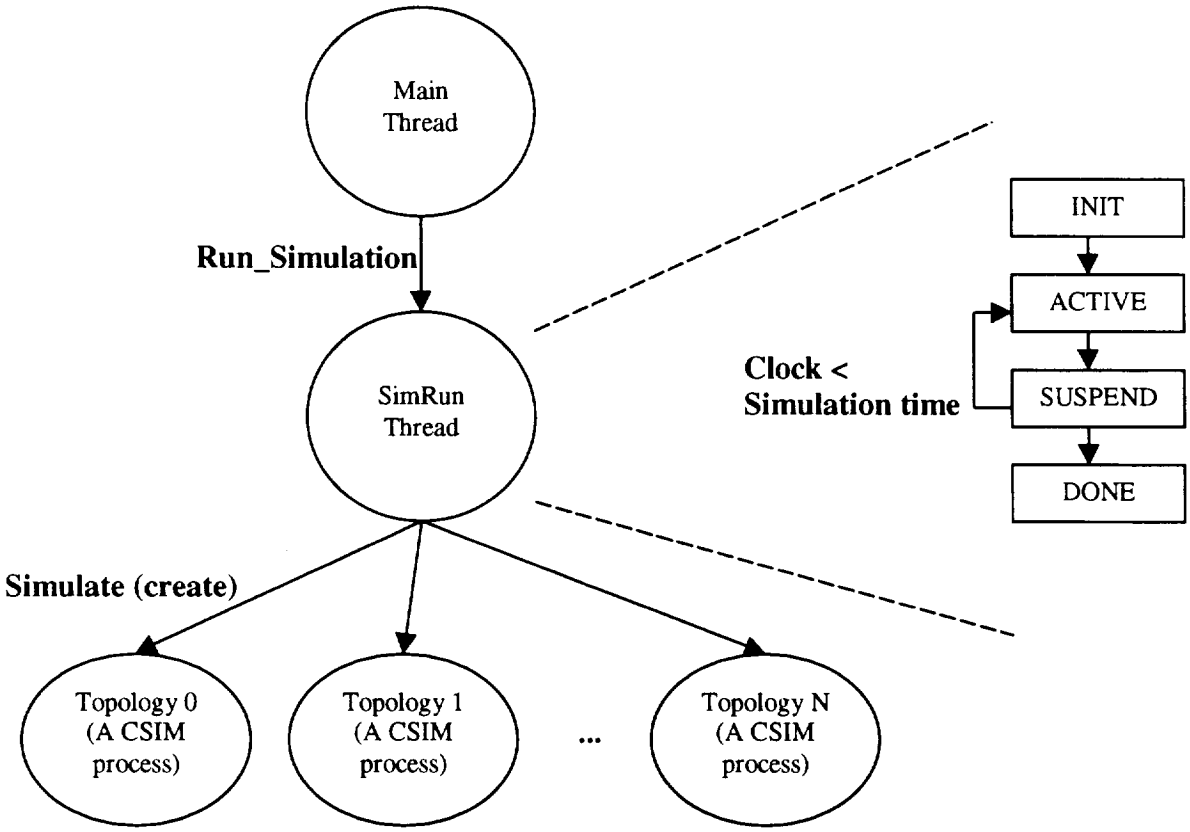

Fig.4. Diagram showing how multiple topologies are simulated under a single clock in a user controllable fashion

We have used Uniform, Priority, Rate based and Adaptive scheduling algorithms at the routers. Any one of these algorithms can be used with each topology in order to study their impact on the outcome. Fig.5. depicts various scheduling algorithms to be evaluated [4].

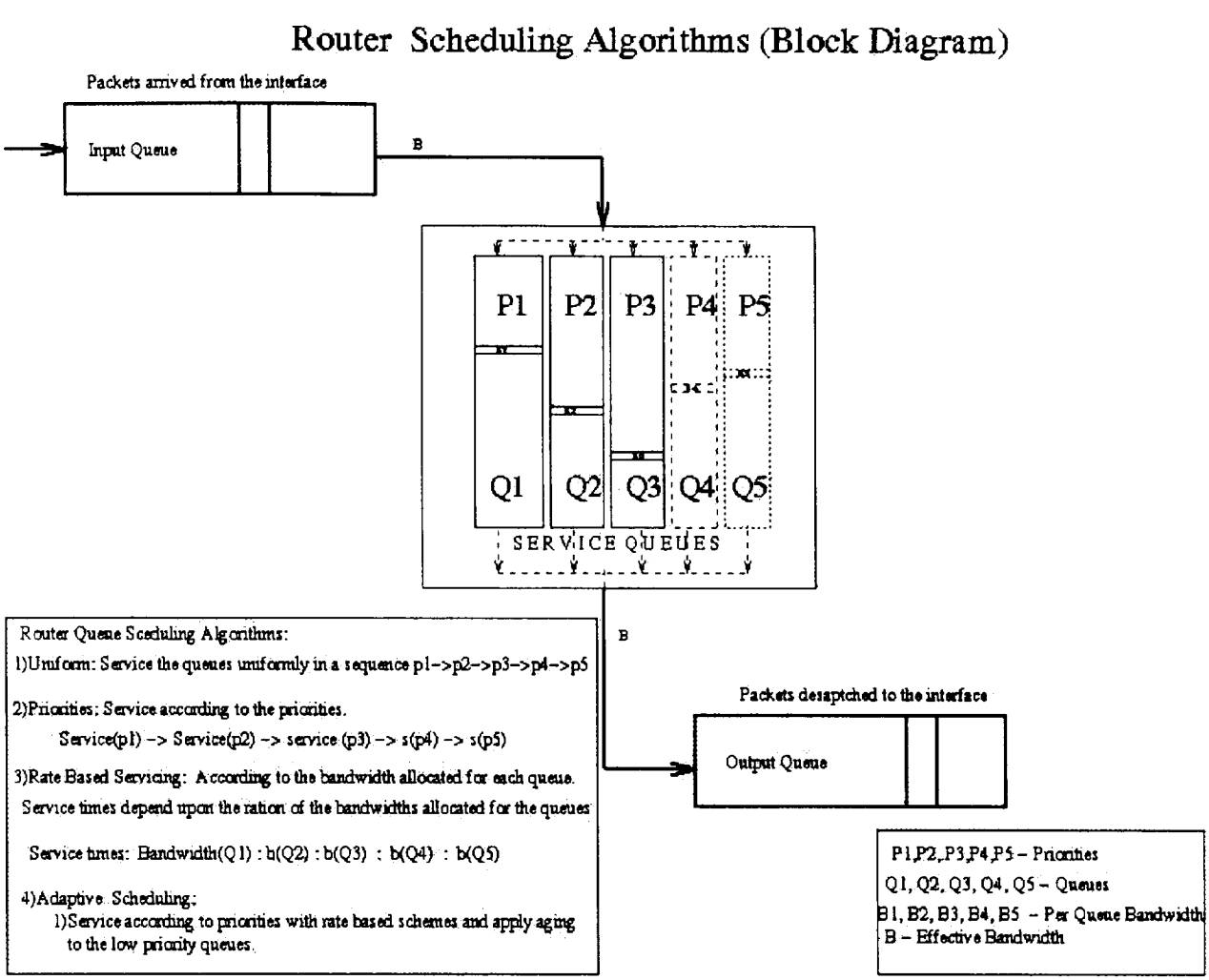

Fig.5. Router Scheduling Algorithms

User Interface

We have decided to implement GUI in Java with the future intention of running it as an applet. This provides platform independence and a wider availability with access via the Web. Some of the latest technology in simulation is to provide simulation, which can be run in a Web browser, thus available to a wider audience.

We have defined a new GUI based language to define network scenarios. This is a new technique with a lot of flexibility to define desired network topology. The user can interactively add a network component to a scenario and edit its parameters. Fig.6 is one such scenario created using our tool. Our name for this tool is 'Scenario Editor'. One can open many such Editors simultaneously to support multiple network scenarios. The network object, which is not visible in the Fig.6, is a 'Connection' object. Each Connection object is defined by specifying source host, intermediate nodes and destination host. The route selection for connection is basically done by mouse clicks along the route. There is no limit on the number of connection objects can exist or the number of hops each connection can run. Fig.9. Shows the interface, which facilitates editing of

parameters of any selected connection object. Normally a connection object parameters include a network id, number of hops it is running in terms of number of intermediate routers, source host id, for each hop corresponding node id and router id, destination host id, message type, priority, distribution with which the connection emanates packets, mean inter arrival time of packets, range from the mean (for uniform distribution), starting time for the connection and duration of a connection existence. A link object is created when the user selects two adjacent nodes. Many links can co-exist between any two nodes. Fig.7. Shows an interface to edit link parameters. Normally a link parameters include network id, link type, source and destination node ids, link length, propagation delay in that link and its capacity. A left mouse click along with shift key press will result in the creation of a node object. Fig.8. Show an interface to edit node parameters. Typically a node parameters include network id, node id, node type, number of routers within that node and for each router, router speed, enqueue time, dequeue time and priority queue sizes and number of hosts connected to them. All the network objects will have unique ids within their domains. We have provided features like serialization of network objects and storing them onto disk as scenario files. These scenario files saved onto disk can be retrieved later and the whole old scenario get instantiated. Here the network scalability is limited by the display size. But one has to compromise on the size in order to view the scenario as such.

Feedback from the simulation server to GUI is shown in a variety of ways. The first such thing is a monitoring system at the server which will continuously monitor the state changes like a priority queue is 80% full or 100 % full and a link is 80 % or 100 % utilized and report to GUI. These state changes are reflected in the scenario editor with different colors. In Fig.6, we can see some nodes are colored in yellow i.e. at least one priority queue at that node is 80 % utilized and some are in red color i.e. 100% utilized. To know the exact utilization user can double click on that particular node, then a histogram is popped up to show different priority queues utilization as show in Fig.12. Likewise a link is also can reflect these state changes. In order to have a comparison base a user can select desired QoS parameters of interest before hand and ask the simulator to send the updates at regular intervals. This type of feedback is shown in the form of strip charts as shown in Fig.10 and Fig.11. Different legends in these stripcharts will reflect different scenarios that are loaded to the server to run under a single simulation clock. In Fig.10, for scenario 0 we have applied uniform scheduling scheme and for scenario 1 we have applied priority scheduling scheme, hence for Q0 (the high priority queue) length for scenario 1 is always far below than the Q0 length for scenario 1, which is logical as priority scheduling scheme will pay more attention towards high priority queues. Whereas in Q4 length the situation has got reversed as the priority scheduling is offering less service to the lowest priority queue, but the uniform scheduling is offering equal service to all priority queues.

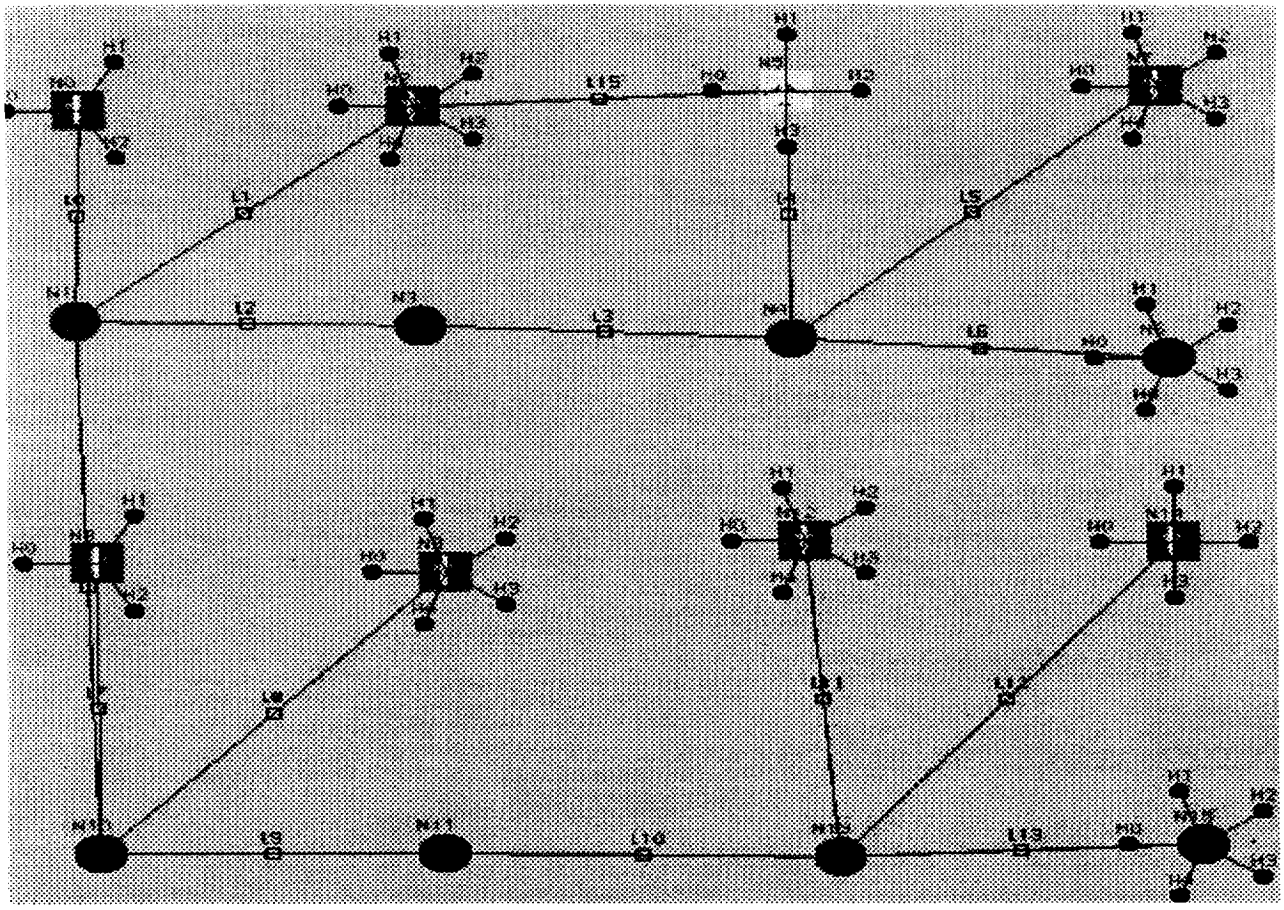

Fig.6. A sample ATN scenario (A big circle being a Ground Router Domain (RD), A Flight symbol being a Mobile Router Domain, a line with a diamond at the middle being a Sub-network and a small circle being a Host within a RD)

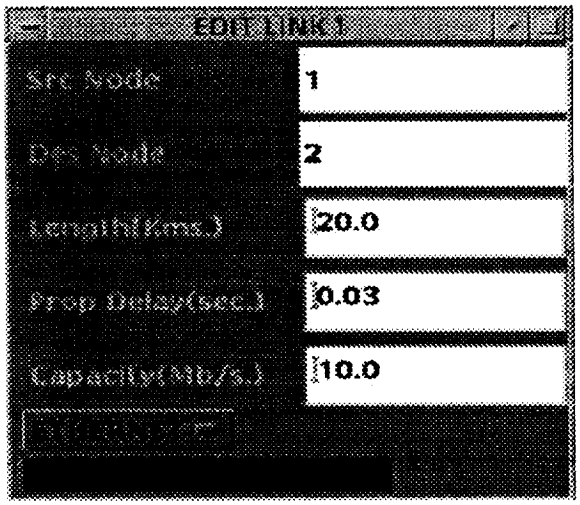

Fig.7. An Interface to enter/edit a Link (Sub-network) parameters

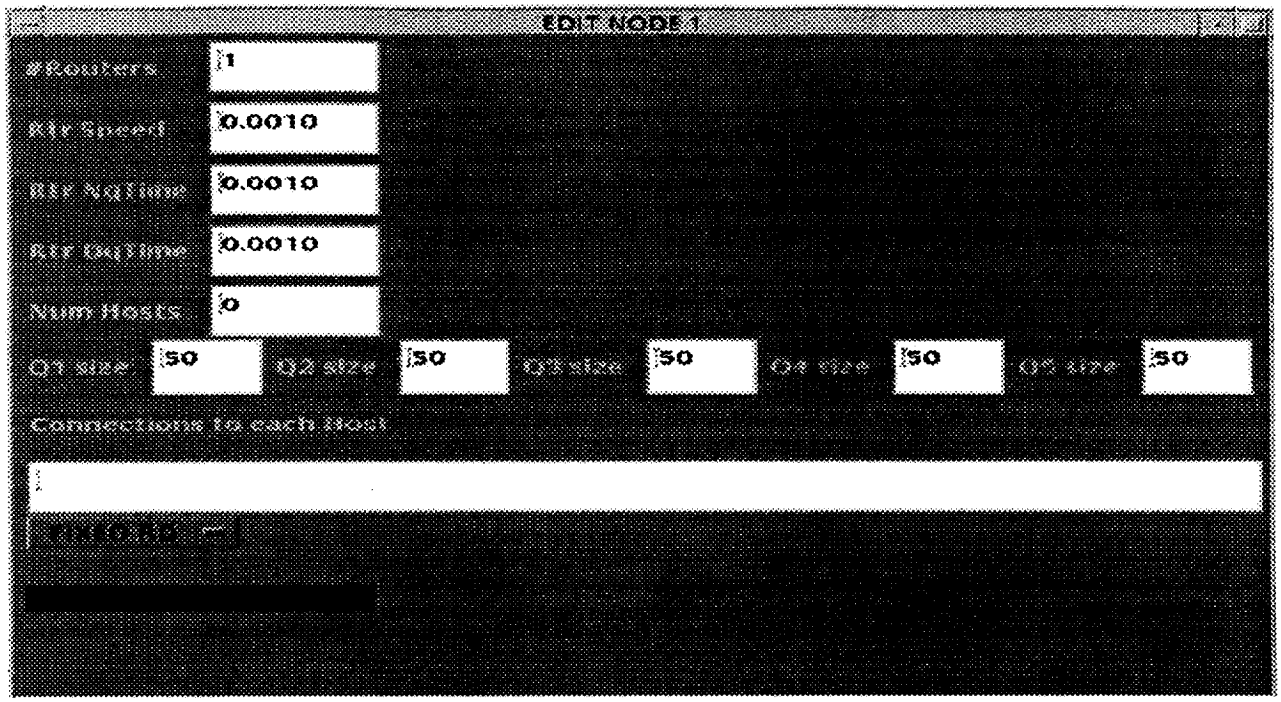

Fig.8. An Interface to enter/edit a Node (RD) parameters

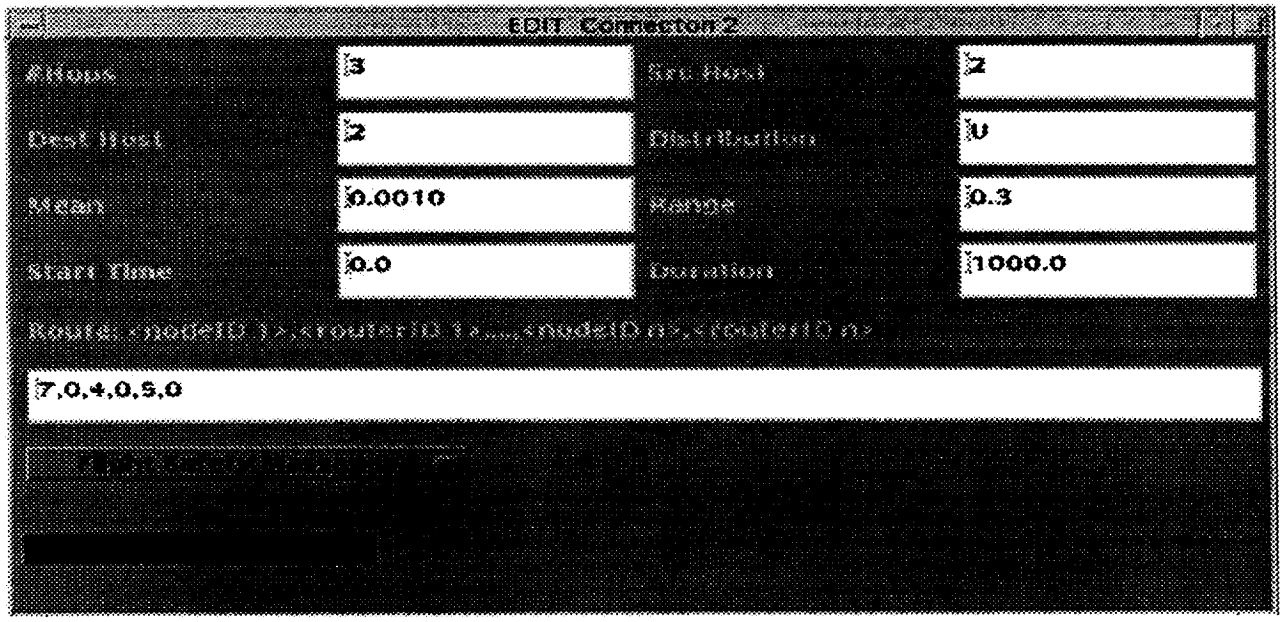

Fig.9. An Interface to enter/edit a Connection parameters

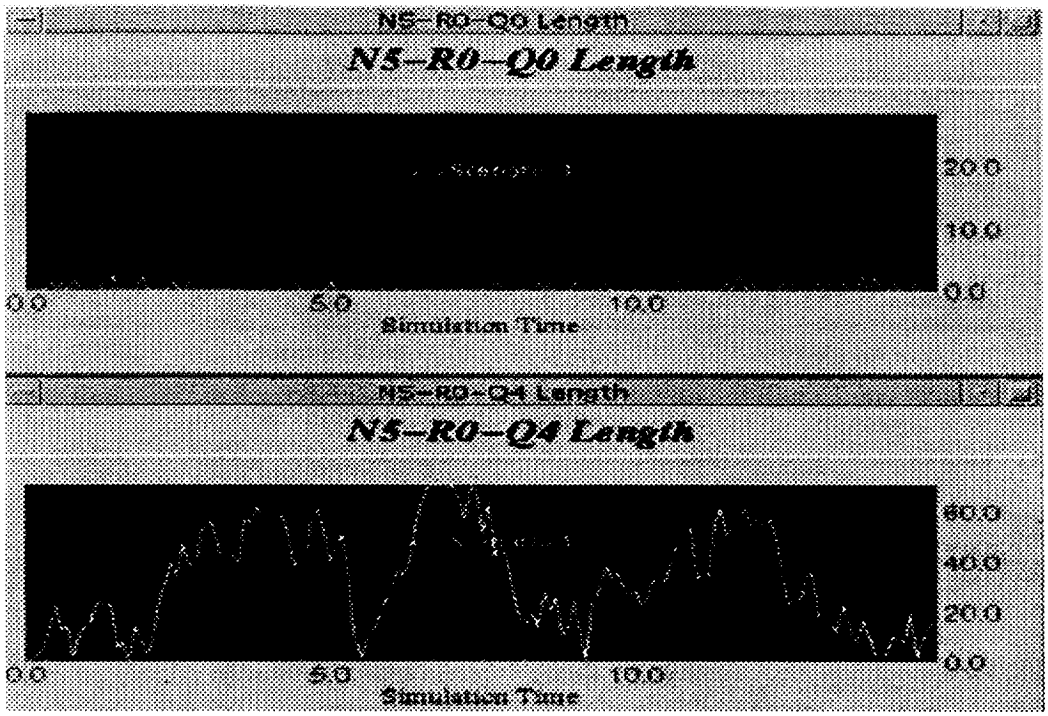

Fig.10. A sample online Strip chart Display (Feedback for a user selected parameter)

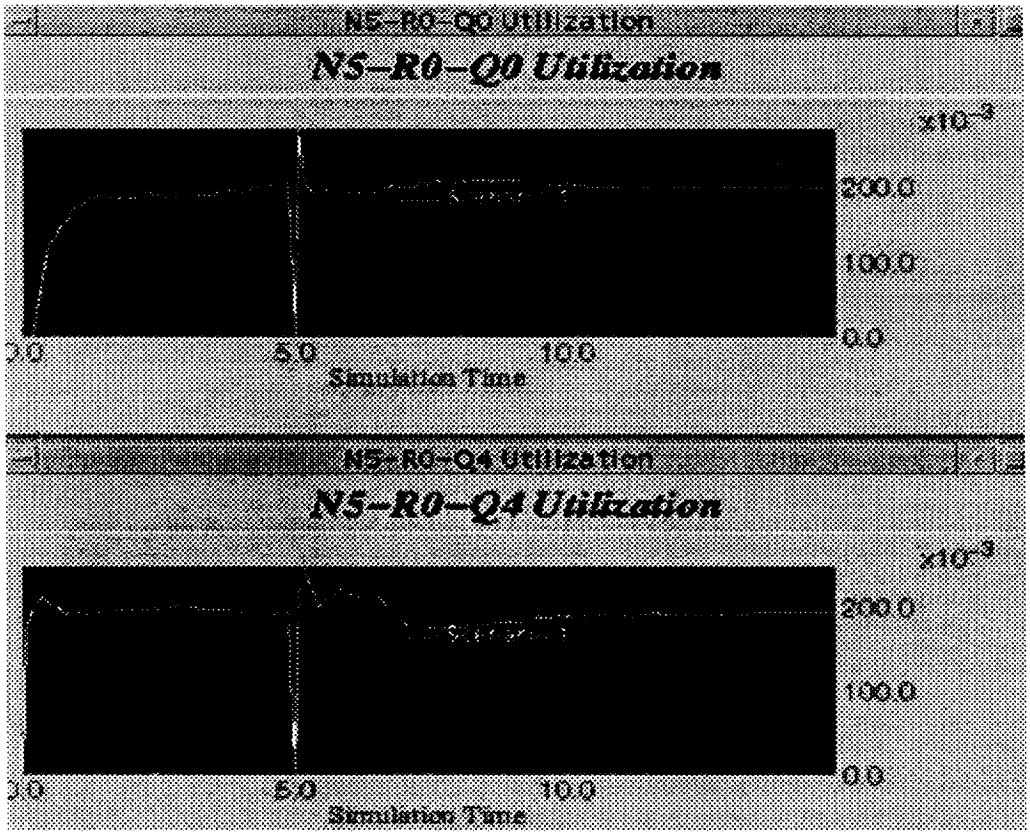

Fig.11. A sample online Strip chart Display (Feedback for a user selected parameter)

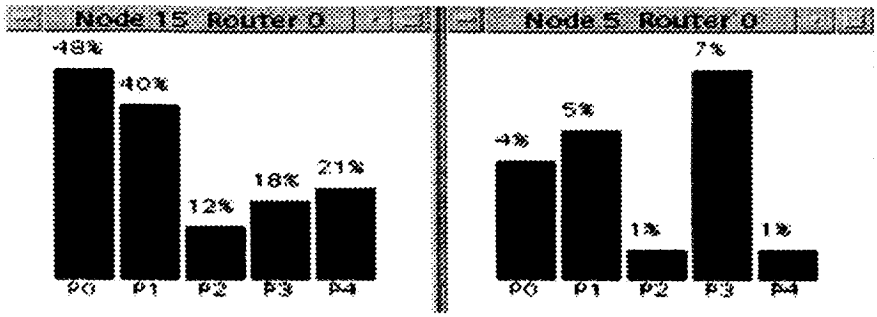

Fig.12. A sample Histogram (A double click on the scenario display will retrieve this information from simulation server)

V. Verification & Validation

There is no actual data available for us to validate the results from our model. But the correctness of the model is being observed by tracking individual packets and comparing the results with the pre-defined routing paths. The correctness of the scheduling algorithms is observed by logically comparing the output patterns from routers that are operated with different scheduling algorithms.

VI. Performance and Results

As a matter of fact achieving good performance should be the ultimate goal for any complex models like ATN. Because ATN is not a simple local phenomenon, but it is rather a global network. So any model trying to simulate ATN should allow it to scale to any size. We have scaled the model to such an extent that only memory can limit the size of the model. All the Data structures we have used to store all these objects can grow dynamically with the size of the model. Hence we can say our model did achieve this requirement. Table 3. Shows how the complexity of a scenario effects the simulation time. Here the complexity is defined in terms of number of links, nodes and connections. As we keep on increasing the complexity at some point, i.e. at the last row, to simulated 20 simulation time units the simulator is taking 17.18 seconds of CPU time.

Mobile RDs	Ground RDs	Links	Connections	Sim-time (sec.)	CPU Time (sec.)
5	2	7	12	20	2.43
10	4	16	25	20	5.54
20	6	25	40	20	9.39
40	8	40	60	20	17.12

Table 3. Complexity of the scenario Vs. CPU time to run a model for 20 simulation seconds

As far as the results from this model are concerned, We haven't done any model studies as such. But using this simulator one can do a lot of capacity studies. The simulator will generate a lot of details that one might be interested in. At each router or a host or a link, one can know about the QoS parameters such as service time, utilization, throughput, waiting time and response time. An exhaustive list of results is provided in Appendix – D.

VII. Status and Future Work

Right now both the simulator and GUI are ready to be used on Unix platforms. In future this can be as well ported on to other platforms as CSIM18 has cross platform capability. We have started designing this model with certain assumptions on the real ATN, so our model is deprived of some of the aspects of ATN and still need to incorporate many of them. Apart from the requirements, the model needs to undergo some design changes for performance reasons. At the GUI side a lot of error checking has to be done in order to prompt the user if something goes wrong. There are lot more priority messages than we have expected, hence we have to include them. Mobile sub-networks have to provide service based on the message priority, which is to be incorporated. Path authorization schemes have to be implemented to restrict message flow only through authorized paths. Broadcasting capability has to be added to the mobile sub-networks. At the sub-networks, we have to provide segmentation/reassembly to allow the conveyance of large packets greater than the sub-network's internal packet size. We have to include the overhead coming from security schemes to avoid internal and external threats to ATN. This model should undergo vigorous VV&A process.

VII. References

[1] ATN standards and Recommended Practices (SARPs), Sub-Volume 1, Introduction And System Level Requirements, Version 2.2, 16 January 1998, http://www.fans-is.com/atnpanel.htm#SARPs

[2] ATN standards and Recommended Practices (SARPs), Sub-Volume 5, Internet Communications Service, Version 2.2, 16 January 1998, http://www.fans-is.com/atnpanel.htm#SARPs

[3] Overview of the ATN Project, Wendy H. Bretton, August 1997

[4] A proposal for Benchmarking the Future Automated ATM System, Kurt Maly, A. Gupta, S. K. Mynam, http://www.cs.odu.edu/~btu/ATN/index_atn.html

[5] Draft ATN SARPs and Guidance Material, Version 0.0, 30 August 1994

[6] CSIM18 (C++ Version) User Guide, http://www.mesquite.com/document/html_cpp/01intro.htm

[7] Graph Package, http://www.sci.usq.edu.au/staff/leighb/graph/Top.html

APPENDIX – A

Acronyms

AOC (Aeronautical operational Control): Communication required for the exercise of authority over the initiation, continuation, diversion or termination of flight for safety, regularity and efficiency reasons.
APC (Aeronautical Passenger Communication): Communication relating to the non-safety voice and data services to passengers and crew members for personal communication.
ATN: The symbol used to designate aeronautical telecommunication network.
ATSC (Air traffic services communication): It includes air traffic control, aeronautical and meteorological information, position reporting and services related to safety and regularity of flight.
CLNP (Connectionless network protocol): The protocol responsible for forwarding packets through the ATN Internet communications service.
ERD (End routing domain): A routing domain (RD) that only routes protocol data units (PDUs) from/to its own RD.
PDU (Protocol Data Unit): A unit of data transferred between peer entities within a protocol layer.
QoS (Quality of service): Information relating to data transfer characteristics used by router to perform relaying and routing operations.
RD (Routing domain): A set of end systems and intermediate systems that operate the same routing protocols and procedures and that are wholly contained within a single administrative domain. A routing domain may be divided into multiple routing sub-domains.
RDC (Routing domain confederation): A set of routing domains and/or RDCs that have agreed to join together. The formation of a RDC is done by private arrangement between its members without any need for global coordination.
SNPA (Sub-network point of attachment): It is a conceptual point within an end or intermediate system at which the sub-network service is offered.
TRD (Transit routing domain): A domain whose policies permit its boundary intermediate systems (BISs) to provide relaying for protocol data units (PDUs) whose source is located in either the local routing domain or in a different routing domain.

APPENDIX – B

Java Simulation Commands (A Scenario Specification Language)

All communication with the simulation server, 'SimSvr', takes place via 'SimCmds'. When the user executes 'Load Scenario', then all the objects pertaining to that scenario are get converted into 'SimCmds' and reach the 'SimSvr'. Basically 'SimCmds' define a protocol between GUI and simulation server. Apart from loading a scenario the 'SimCmds' include setting simulation time and other control parameters, statistics requests and setting desired scheduling scheme. All the 'SimCmds' will be sent to a port where 'SimSvr' is listening.
Format:- <cmd#> <args for that cmd>

<cmd#> - The following is a list of commands that the SimSvr understands.

0	- Add a Node
1	- Add a Link
2	- Add a Connection
6	- Set Simulation Time
10	- Set Scheduling Scheme
15	- Execute the Simulation
16	- Establish a socket for statistical output for a particular topology
19	- Print the content of a particular topology structure
20	- Tell the SimSvr to exit

<args for the Cmd>
Arguments for a adding a Node would be:
<net id> - index of the network or topology (start at 0)
<node #> - node number (start at 0)
<node type> - mobile or ground based RD
For each router list the following as args.
<rtr NQ time> - time router takes to enqueue items in msec.
<rtr DQ time> - time router takes to dequeue items in msec.
<Qsize 1> - maximum size of priority queue 0
<Qsize 2> - maximum size of priority queue 1
<Qsize 3> - maximum size of priority queue 2
<Qsize 4> - maximum size of priority queue 3
<Qsize 5> - maximum size of priority queue 4
<#hosts> - number of hosts at this node
For each host
<no. Connections> - number of connections host supports

Arguments for adding a link would be:
<net id> - index of the network or topology
<link type> - defined as follows

0	Mode-S
1	SATCOM
2	VHF
3	ATM

4 T1
5 Ethernet

<src node id> - node number representing source node of link
<dest node id> - node number representing destination node of link
<length> - length of the link in KM
<capacity MB/s> - capacity of link in MB/sec

Arguments for adding a Connection would be:

<net id> - index of the network or topology
<#hops> - number of nodes the connection traverses, including source & destination
<src host id> - node number representing the connections start point

For each node the connection travels through list.
<hop id> - node number
<rtr id> - index of router (start with 0)

<dest host id> - host number representing the connection's end point
<message type> and <priority>, both are defined as follows:
0 Urgent Communications
1 Flight Safety Messages
2 Weather Broadcast
3 Flight Regularity Messages
4 Aeronautical Info Service Messages
<distribution> - distribution to use
<mean> - mean arrival rate of packets (double)
<range> - mean deviation (double)
<start> - simulation time at which to start the connection
<duration> - number of simulation clock ticks the connection should last

Arguments for setting the simulation time:
<time> - double representing how long you want the simulation run

Arguments for setting the scheduling scheme for queue service:
<net id> - index of the network
<method #> - defined as follows:
0 service queues uniformly in sequence
1 service queues according the priorities
2 service queues according to the queues bandwidth
3 service queues by priority applying bandwidth & aging

Arguments to execute simulation:
None

Arguments for establishing output socket:
<net> - index of the network

Arguments for printing content of a network structure:
<net> - index of the network

Arguments for telling the simulation server to exit:
None

APPENDIX – C

Execution Procedure

Step 1: Start Simulation Server
% cd /home/simulati/atn/bin
% SimSvr <port_num (any arbitrary port)>
e.g. % SimSvr 3400

Step 2: Start Java GUI
% cd /home/simulati/atn/classes
% source ../ATN.cshrc
% java atn <server's hostname> <port_num (same as the one given in step 1)>
e.g. % java atn pitfall 3400 (If the SimSvr is running on host 'pitfall')

Step 3: Select 'Start Server' option from 'Simulation' menu

Step 4: Select 'Create Scenario' option from 'Configure' menu
choose 'new' option to create a new scenario
choose 'open' to view/edit an already created scenario

Step 5: Follow the help command to edit/create scenario

Step 6: Load the scenario to SimSvr by selecting the 'Load Scenario' option form 'Scenario' menu

Step 7: Select 'Set Scheduling' Scheme option from 'Scenario' menu
choose one from Uniform, Priority, Rate Based and Adaptive

Step 8: Repeat from step 4 to 7 for multiple scenarios

Step 9: Select 'Choose Statistics' option from 'Configure' menu of main atn interface
choose any QoS parameter (Service Time, Utilization, Throughput, Length, Response Time) of interest for a priority queue

Step 10: Repeat above step to view multiple statistics simultaneously

Step 11: Select 'Run Simulation' option from 'Simulation' menu of main atn interface

Step 12: Double click on any node/link to see the Utilization during the simulation run

Step 13: Select 'Report' option from 'Display' menu of main atn interface to view the final CSIM report when the simulation run is finished

Step 14: Use 'Exit' option from 'Simulation' of main atn interface in order to quit the simulation

APPENDIX – D

Statistics

The following is the exhaustive list of statistics our simulator can provide. Statistics can be categorized into two varieties: 1. Application calculated values, which are the result of book-keeping done at different stages of the simulation. 2. CSIM report, which can be obtained as a result of using CSIM objects.

1. Router:- max. number of queues=5, for each queue following attributes are being observed.

Application calculated values,

1. number of packets received (no_rcvd)
2. number of packet dropped (no_drop)
3. current qsize (count)

CSIM18 qhistogram report,

1. Mean service time per request(serv)
2. Mean Utilization (busy time/elapsed time) (util)
3. Mean Throughput rate (completions per unit time) (tput)
4. Mean number of packets waiting or in service (qlen)
5. Mean time at queue (waiting time and service time)(resp)

2. Host:-

CSIM18 host histogram report,

1. Mean service time per request (serv)
2. Mean Utilization (bust time/elapsed time) (util)
3. Mean Throughput rate (completions per unit time) (tput)
4. Mean number of packets waiting or in service (qlen)
5. Mean time at queue (waiting time and service time) (resp)

3. Link:-

Application calculated values,

1. Total number of bytes received (bytes_rcvd)

CSIM18 storage report,

1. number of storage currently available (avail)
2. number of processes currently waiting at store (qlength)
3. sum of requested amounts for Store (request_amt)
4. time-weighted sum of requesters for Store (number_amt)
5. busy time-weighted sum of amounts for Store (busy_amt)
6. waiting time-weighted sum of amounts for Store (waiting_amt)
7. total number of requests for Store (request_cnt)
8. total number of completed requests Store (release_cnt)
9. number of queued requests at Store (que_cnt)
10. time at Store that is spanned by report (time)

4. Connection:-

Application calculated values,

1. number of packets dropped (no_dropped)
2. number of packets received (no_rcvd)
3. number of packets offered (no_offered)
4. number of bytes received (bytes_rcvd)
5. cumulative delay (cum_delay)
6. max. delay (max_delay)
7. min. delay (min_delay)