# TECHNICAL RESEARCH REPORT

Integrating NASA Missions into the Internet using Commercial Satellite Constellations: Implementation of ATM-Based Cell-Relay Protocol Providing Support for Mission Integration

*by Sreenivas Ramaswamy*

**CSHCN T.R. 2000-9**
**(ISR T.R. 2000-27)**

*The Center for Satellite and Hybrid Communication Networks is a NASA-sponsored Commercial Space Center also supported by the Department of Defense (DOD), industry, the State of Maryland, the University of Maryland and the Institute for Systems Research. This document is a technical report in the CSHCN series originating at the University of Maryland.*

**Web site http://www.isr.umd.edu/CSHCN/**

Independent study report
Spring 1999

Integrating NASA missions into the Internet using commercial satellite constellations

# Implementation of ATM-based cell-relay protocol providing support for mission integration

**Sreenivas Ramaswamy**

Integrating NASA missions into the Internet using commercial satellite constellations:

# Implementation of ATM-based cell-relay protocol providing support for mission integration

Sreenivas Ramaswamy

**Abstract**

Near-earth spacecraft are considered to be very LEO satellites. The commercial constellations to be tested range from high-density LEOs to GEOs. We have presently developed a complete baseline simulation model that uses basic algorithms for all of the test parameters. We are working now on developing individual test modules for each parameter, which will then be plugged onto the baseline model and evaluated for each satellite system. The baseline model itself will be reinforced with the Opnet ATM suite to replace the current quasi-MAC protocol.

Following the successful implementation of basic algorithms for each of these test cases (shortest path routing, hard handoff, beacon monitoring, file transfer, deterministic packet transmission, single-priority FIFO queuing), we are currently working on replacing the MAC protocol with the ATM suite. The advantages of this approach are two-fold: first, it allows us greater flexibility in including multiple service classes, traffic types and priority schemes. Also, it creates a well-defined network/switching layer upon which TCP/IP and UDP/IP can be evaluated.

## 1. Background

There are currently two independent research thrusts that involve communications support for NASA missions:

1. One is the "satellite as a node" project at NASA GSFC. The objective of this project is to address networking and protocol issues that will allow future space missions to be additions to a global Internet, so that authorized users anywhere can directly access on-board instrumentation and databases to conduct experiments and retrieve payload.
2. The other project is the use of commercial satellite constellations to support payload transfer and TT&C operations for NASA space missions, to allow for increased user access to NASA missions and lower demands on the NASA space network.

We are focusing on addressing both these problems as a single research problem. To do so, we need to address the following two issues:
- physical and data link interfaces that are compatible with existing terrestrial infrastructure and which are to be deployed on commercial satellites,
- network and transport layer protocols for data communication, that are compatible with current and proposed gateway interface protocols and Internet protocols.

Our goal is to develop a full-fledged theoretical and simulation model for analysis of the problem. A model for each component must be built separately in the form of a baseline model and a set of test cases. An important baseline problem is that of protocol support. That is, it is common to all the different test scenarios that involve different satellite constellations, applications, traffic models and routing strategies. In each of these test cases, the protocol stack must be well established and proven to be able to support multiple classes of services and multiple applications, so that the Internet can be extended across the space link.

## 2. Introduction

The commercial network is based on Orbital Sciences Corp.'s ORBLINK® constellation of 7 satellites in MEO orbit, at an altitude of 9,000 km. above the surface. Each satellite possesses sufficient OBP intelligence to make routing and bandwidth allocation decisions, and inter-satellite links are used between the satellites to form a ring network in space.

The ground network is made up of three ground-space gateways, each of which is connected to three or more terrestrial network gateways. The network gateways serve as sources of data traffic from terrestrial LANs and corporate WANs. The ground-space gateways are responsible for satellite PAT procedures and for coordinating handoff between satellites. The ground-space gateways are also responsible for protocol support to ensure that end-to-end QoS for selected services is not compromised. Initially, only the ground-space gateways will be able to communicate with the satellite network over the UDL. Time permitting, we will also explore the implementation of Mobile User Link (MUL) connectivity to allow network gateways and mobile users to communicate directly with the satellites.

The ATM suite in Opnet 4.0 is made up of three modeling layers: the ATM layer, the AAL layer, and the IP interface layer. The general model architecture of the ATM model in Opnet can be described as follows. ATM packets that are received over the network are accepted by an ATM Translation module. This module translates VPI/VCI values for outgoing cells and forwards cells to appropriate modules. The ATM Management module invokes ATM processes to handle call signals and routing messages. An ATM Switch module performs cell switching and switch buffer dequeuing.

In our Opnet model, each satellite is modeled as an ATM switch, and hence is composed of these four modules, suitably modified to our specifications. For end-to-end operations, the ISS and ground terminals include, in addition to these four modules, the following: an ATM Layer which interfaces the switch to AAL layers; AAL Layer modules which handle AAL signaling to establish and release AAL connections; and an IP Interface module that transport IP datagrams across the ATM network.

In this manner, we split the ATM model over the entire network, with the satellites acting as ATM switches, the end users modeled using the entire ATM layered model. The interesting aspects of this model include the effect of propagation delays, inter-satellite links, handoffs, on-board queuing and bandwidth splitting between commercial traffic and ISS-related traffic. The goal of this study is to identify these and more parameters that affect the end-to-end performance of the ATM layer over high-density satellite networks. Future work will then focus on improving the performance by avoiding some of these problems completely, or solving them through increased complexity at the network (IP) or transport (TCP) layer.

The performance parameters that need to be addressed include:

- COVERAGE : Percent of time that data could be transmitted to the ISS via the commercial satellite system (this includes Static & Dynamic coverage and the effect of Inter Satellite Links).
- THROUGHPUT : Maximum amount of information that can be exchanged between constellation & ISS, based on service availability and the per channel data rate.
- QUALITY-OF-SERVICE : Level of confidence for the reliable delivery of information to NASA users: Link quality (BER), Link Availability, Connectivity.
- ANTENNAS & TERMINALS : Antenna & earth terminal characteristics with respect to required link quality. It would be necessary to have an antenna design well suited for covering both LEO vehicles and terrestrial traffic.

## 3. Present model

The baseline model that is complete is made up of the following components. The network model is as shown below.

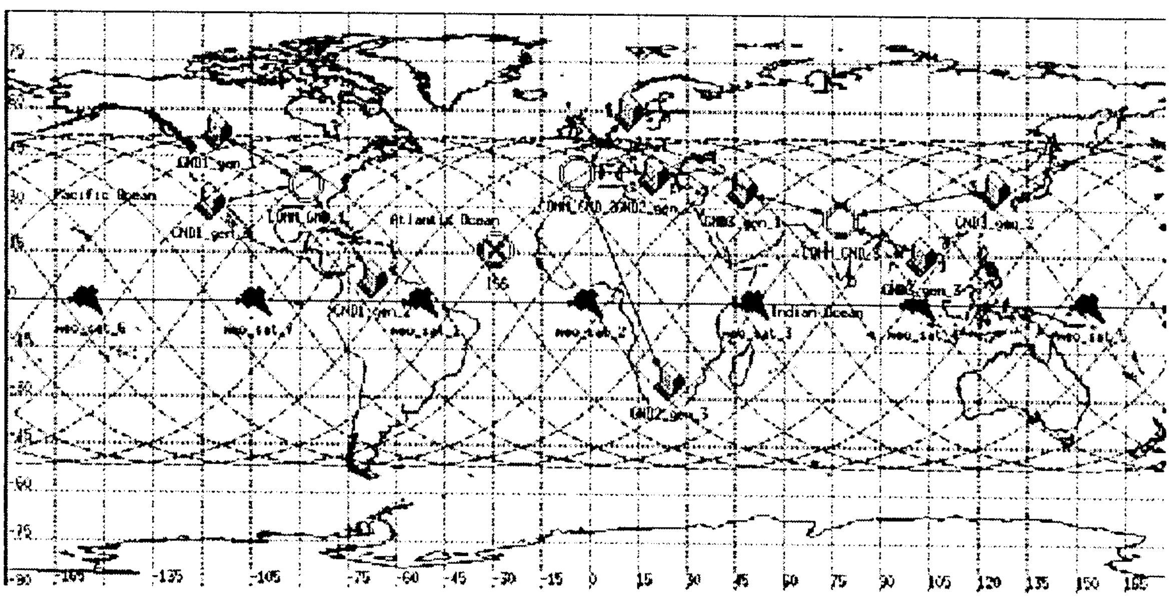



*Fig 1. Opnet network model*

The network model is made up of 7 satellite in MEO orbit (meo_sat_1 through 7), 3 ground stations (comm_GND_1, 2, 3), a number of network gateways (GND1_gen, etc.), and the ISS itself.

## ISS

The ISS is currently modeled as a simple **ISS_transmit** traffic generator. After a random idle period, it creates a file whose size is uniformly distributed. The file is then divided into fixed-size packets that are created and transmitted deterministically. Destination addresses for each file are determined randomly from among the nine end-user terminal addresses. All packets within a file are sent to the same end terminal. No priority or service classes are implemented. The **queue_sat** module performs simple FIFO queuing, with a packet service time that is chosen to ensure proper flow control. There are infinite capacity transmit queues on ISS. Packets are transmitted only if the strength of the beacon signal received from any MEO is above a threshold value that is a simulation attribute.

Continuous monitoring of beacon signal strengths from 7 MEO satellites ensures correct operation of Pointing, Acquisition and Tracking (PAT) subsystem on-board the ISS, as shown in the node model. The **ISS_beacon_tx** module continuously broadcasts beacon signals to allow other nodes in the network to locate the ISS. Beacon signals that are received from the MEO satellites are processed by the **ISS_beacon_rx** module. The seven radio receivers measure the signal strength that of the beacon that is received from each satellite. The result is made available to the **queue_sat** module to determine if the ISS can transmit packets.

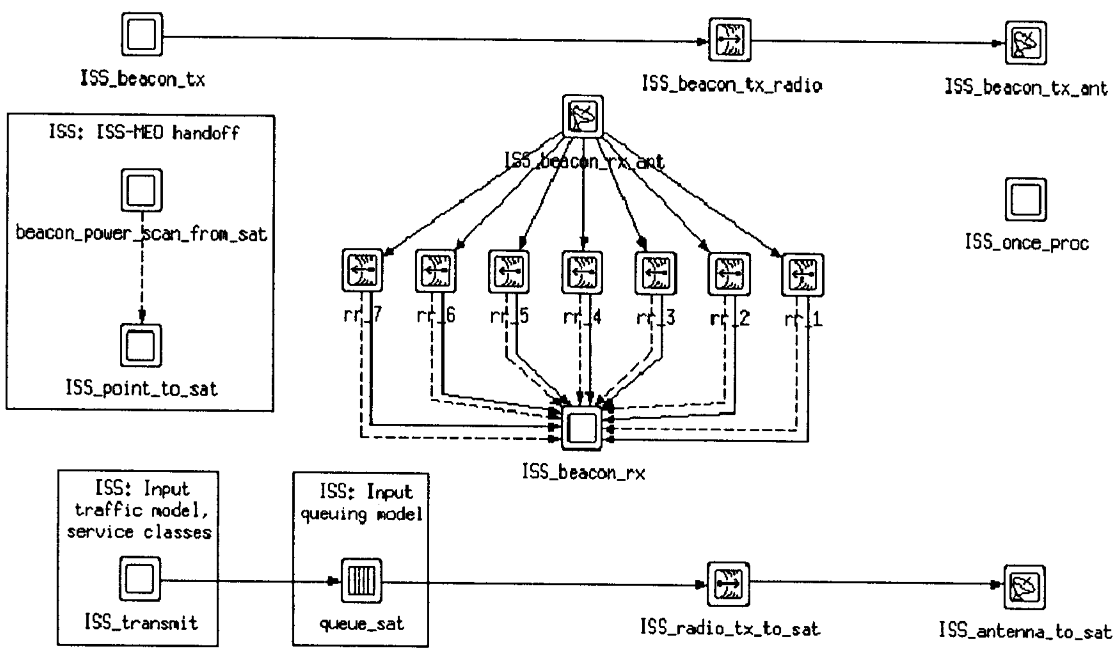



*Fig 2. Opnet node model for ISS*

The **ISS-MEO handoff** modules perform handover of the ISS transmit antenna. Based on the received signal strengths, the **ISS_antenna_to_sat** is handed off between satellites. Handoff on-board the ISS is performed as hard handoff (break-before-make).

**ISS_once_proc** is responsible for initializing state variables, model attributes and process attributes, and maintains the integrity of the node model over multiple simulation runs.

Moving the MAC layer to ATM will allow us to support multiple services in addition to the present file transfer -- video, long-duration connections, multicast, high-priority data, etc. Protocol support at ISS will ensure that QoS requirements are met for each service type. Complex input traffic models will be used to model the distribution of different service applications.

**MEO satellite**

The MEO network is currently made up of 7 intelligent satellites, capable of OBP activity -- queuing, routing and handoff. Each satellite maintains continuous connections with its two adjacent satellites, and all 7 satellites form a ring in equatorial orbit at 9000 km. altitude. The **meo_point_to_meo** module checks and maintains the connections between adjacent satellites. Each MEO satellite has multiple transmit-receive pairs to adjacent satellites, the ISS, and the three ground stations. These tx-rx pairs are identified by the transmitter and receiver modules that feed into the queuing modules (**rx_next_sat**, **rx_prev_sat**, etc.) and receive data from the routing and processing module.

When a satellite receives a packet, it identifies it as belonging to commercial or ISS traffic. Commercial traffic is fed into the **meo_pk_queue** while traffic to or from the ISS is received in the **iss_pk_queue**. A FIFO queuing discipline is used in both queuing modules, because the generated traffic from commercial end stations and the ISS are composed of a single priority class. Additions to this model will include a priority-based queuing scheme based on QoS specifications for packet streams.

The **meo_proc** processing module then performs empirical network-layer (or MAC layer) shortest-path routing and forwards the packet to next-hop satellite or destination ground station. This is done based on the value in the "destination address" (see Packet Formats section) field. From the destination address of the end-user terminal, the satellite determines the closest ground station to the terminal. Continuous location monitoring allows the satellite to know if it is currently in line-of-sight of the destination ground gateway. If so, the satellite downloads the packet to the destination ground gateway. Otherwise, it forwards the packet to one of its neighboring satellites based on Dijkstra's shortest-path algorithm, or destroys the packet if its lifetime is exceeded. The operations that every satellite node performs include MAC-layer echo cancellation, address resolution, hop-count based lifetime control, and shortest-path routing.

The PAT subsystem performs continuous monitoring of ISS beacon signal and beacons from three GND stations to ensure correct operation. The **beacon_tx_proc** module on each satellite continuously transmits low bit rate beacon packets to the ground gateways and to the ISS. Beacon signals that are received from the ISS and each of the three ground gateways is analyzed for signal strength. The **MEO-GND handoff** subsystem monitors the signal strengths and implements hard handoff between GND stations.

**Once_proc** is responsible for initializing state variables, model attributes and process attributes, and maintains the integrity of the node model over multiple simulation runs.

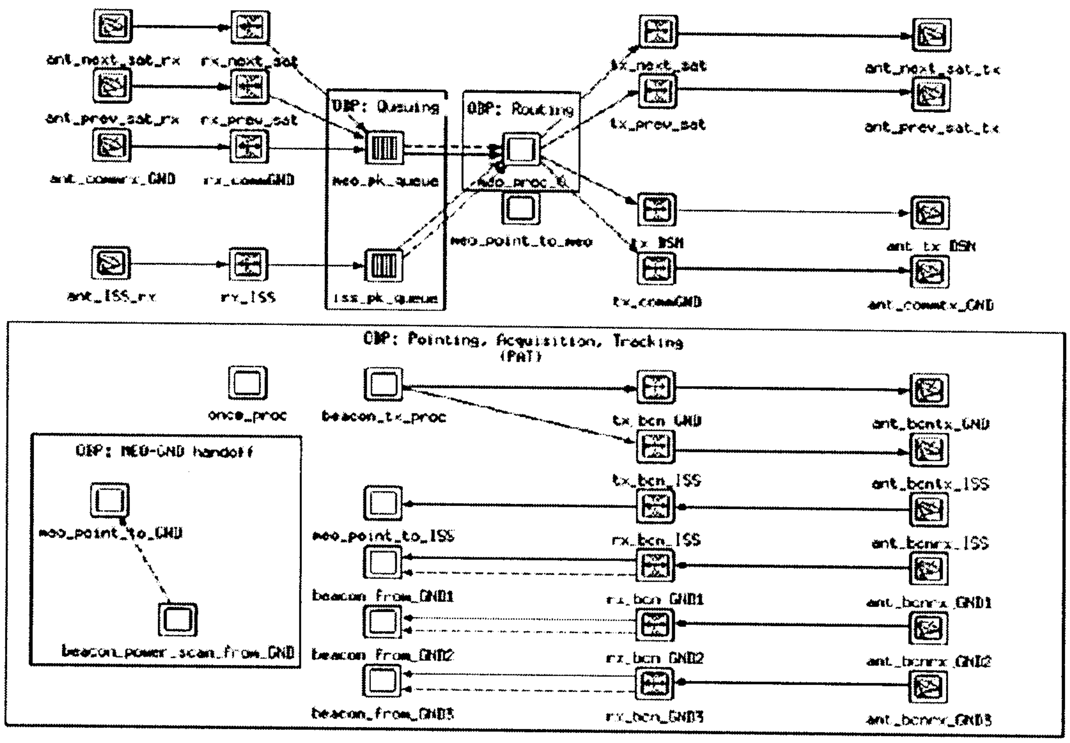



*Fig 3. Opnet node model for MEO satellite*

Once the satellites are modeled as ATM switches, IP protocol implementation at satellite nodes allows us to perform IP-level routing. The IP-over-ATM problem is already a well-known problem with many research efforts addressing various parts of the problem. At satellite nodes, IP will be limited to IP-routing component. No ARP is recommended over the satellite network, and IP-Encapsulation is not needed because the network layer is highest layer at the MEOs.

**Ground station**

The simulation model currently has 3 ground stations that continuously monitor the movement of the MEO satellites to ensure correct PAT operation. Each GND station receives, from ground terminals, commercial traffic to be transmitted over satellite to other ground terminals. It also receives ISS traffic to be transmitted to ISS. GND stations also receive return traffic from the MEO network that is made up of ISS and commercial traffic. These packets are received by the `sink_rr` receiver module. Received packets are queued at the `sink_queue` to be transmitted to end-users. The `sink` processing module uses an impartial FIFO de-queuing scheme to remove received packets from the queue and send them to one of the three end-user terminals based on the packet's destination address. All three end terminals are connected to the ground gateway using point-to-point links (`pt_0`, `pt_1`, `pt_2`).

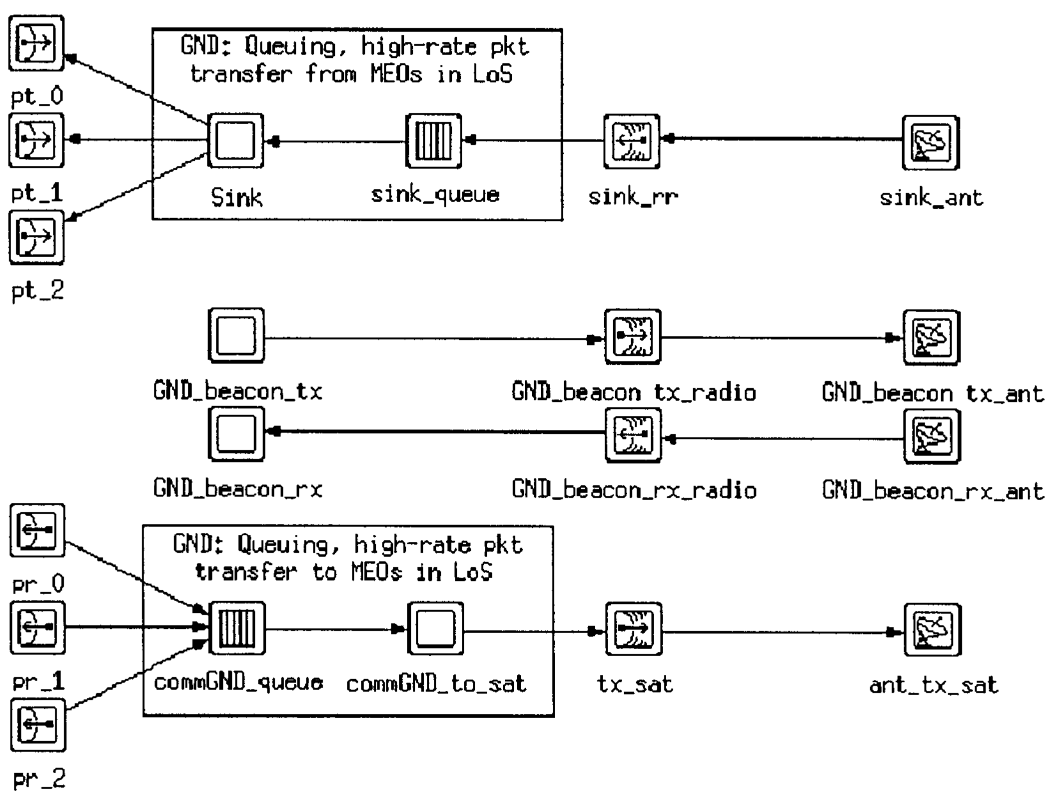



*Fig 4. Opnet node model for GND station*

Point-to-point links are also used to receive data packets from the end terminals. A simple queuing model is implemented at present, with intelligence to initiate high data rate transfer of queued packets to satellite during periods of visibility. The bandwidth is shared equally between ISS packets and commercial packets in the `commGND_queue` module. The `commGND_to_sat` module periodically checks for LoS to any satellite and initiates high rate transfer from the queue to the satellite.

`GND_beacon_tx` and `GND_beacon_rx` modules are responsible for the background beacon tracking operation to ensure that minimal number of data packets are lost due to small and rapidly-changing LoS windows at the ground gateway. The beacon mechanism logically links the ground gateway network with the MEO satellite network.

With IP-over-ATM implemented, IP support at GND stations will provide seamless integration of ground and space segments. IP support will only use IP-Routing component over space segment, IP-Routing and IP-ARP over ground segment. Advanced bandwidth allocation and queuing models can be used to partition available bandwidth between commercial traffic and ISS traffic, with the partition scheme being a test case.

**Ground terminal/Network gateway**

The network model shows 9 ground terminals that are connected to the 3 GND stations (three to each). These terminals can be considered to be network gateways to corporate/local/wide-area networks. Each terminal acts as a source and sink for data traffic to/from other terminals and to/from the ISS. The modules `GND_gen` and `Sink` perform these functions at the network end-user terminals.

A simple FIFO transmit queue is shared by both types of traffic. The receiver queue at network gateways performs segmentation/reassembly, MAC-layer packet sequencing, and duplicate packet detection and discarding. SAR operations are performed based on the packet's sequence number. Packet sequencing operations are carried out using an internal queue called the overflow queue, which stores packets that are received out of order. If a packet's sequence number is less than expected, it is discarded as a duplicate. If the sequence number is greater than expected, it is inserted into the overflow queue and the queue is sorted using a bubble-sort technique. The head of the overflow queue is then checked to see if it is the packet with the expected sequence number. This operation is performed in the `sink_queue` process model.

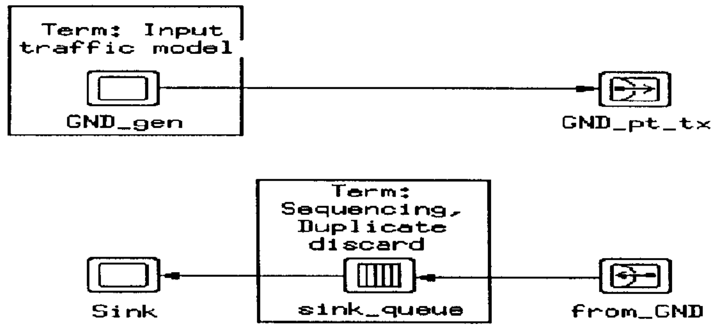



*Fig 5. Opnet node model for ground terminal/network gateway*

Improved traffic models are planned at transmitters to model multiple traffic types for different service classes and QoS requirements. IP (or other network-layer) protocol and basic TCP implementation to provide support for end-to-end QoS guarantees for multiple services. All three IP components (IP-ARP, IP-Encapsulation and IP-Routing) will be implemented. End-to-end statistic collection, average packet delays, packet loss, queue lengths, performance for each service class and traffic type.

### Packet formats

Multiple packet formats are used to model the transfer of information between ISS and users, and between commercial users. The beacon transmission mechanism also employs three different packet formats for GND tracking of MEOs, MEO tracking of ISS and GND, and ISS tracking of MEOs. The following figure shows the simple packet format used between ISS and ground users.


*Fig 6. Packet format used for ISS and ground user data transfer*

The **start_tag** field contains no useful information and may be used in the future for special in-band signaling purposes. **Sequence_number** is used at the receiver terminals for sequencing, duplicate detection and segmentation-reassembly functions. The **hop_count** field is used at ground gateways and the satellite nodes to perform packet lifetime control. The exact value should be computed based on the offered load in the network. The **previous_hop** field is used on the satellites for routing and echo cancellation. Addressing information is carried in the **source_address** and **destination_address** fields.

In this preliminary model, we do not perform any testing of traffic models and so the packets do not contain any information. Thus, the **data_payload** field is empty. However, the packet headers are of importance in the model as they are used to perform the functions of source identification, routing, lifetime control, sequencing, segmentation and reassembly, and duplicate detection and discard. The ATM model requires the use of pre-designed ATM packet formats, and these will be used to implement the ATM network and AAL layers in the simulation model.

## Preliminary results

As expected, the analysis of end-to-end delays, queuing times and loads on the network do not yield any useful information, since the model itself is only a preliminary model to be used as a proof of concept. However, analysis serves to verify the correct operation of the different components in the network. When some input variables are tweaked, we see a variation in some plots that illustrate the correct operation of the model.

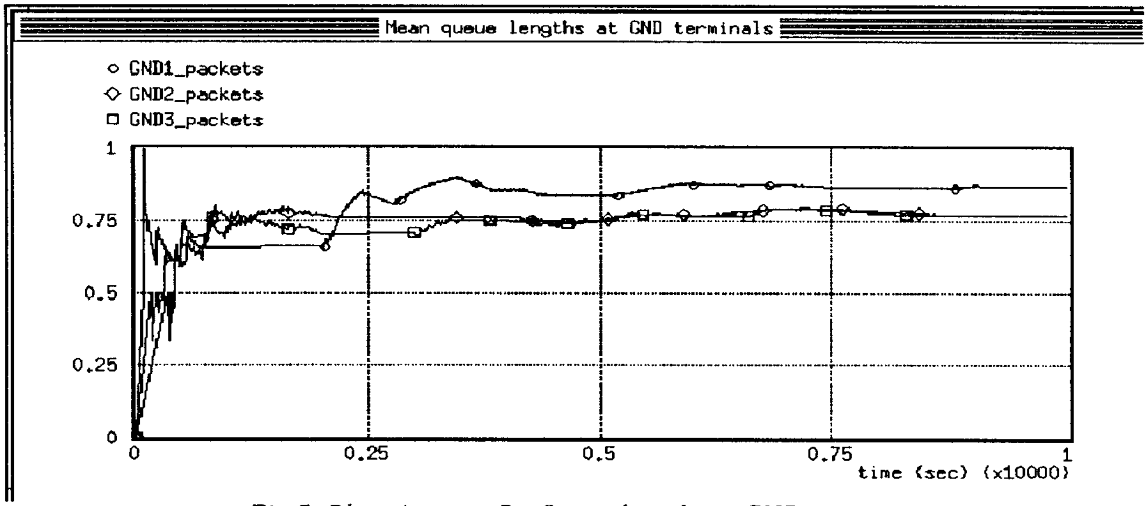



*Fig 7. Plot: Average Rx-Queue lengths at GND stations*

The chart in fig. 7 plots the average receive-queue lengths at each of the three GND terminals, and shows that traffic flows in the network are symmetrical. Receive queue lengths at each GND station are similar, as expected from a correctly working model. Fig. 8 plots the average queuing delay at each GND station.

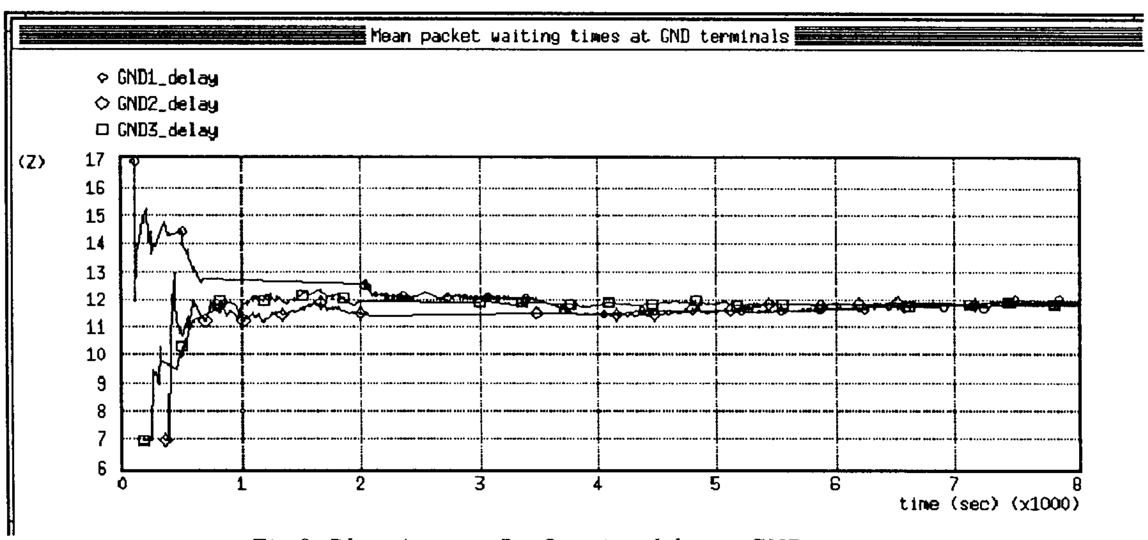



*Fig 8. Plot: Average Rx-Queuing delay at GND stations*

The following chart, fig. 9, plots the queue length over a fixed time interval over selected satellites. It shows the variation in load of each satellite.

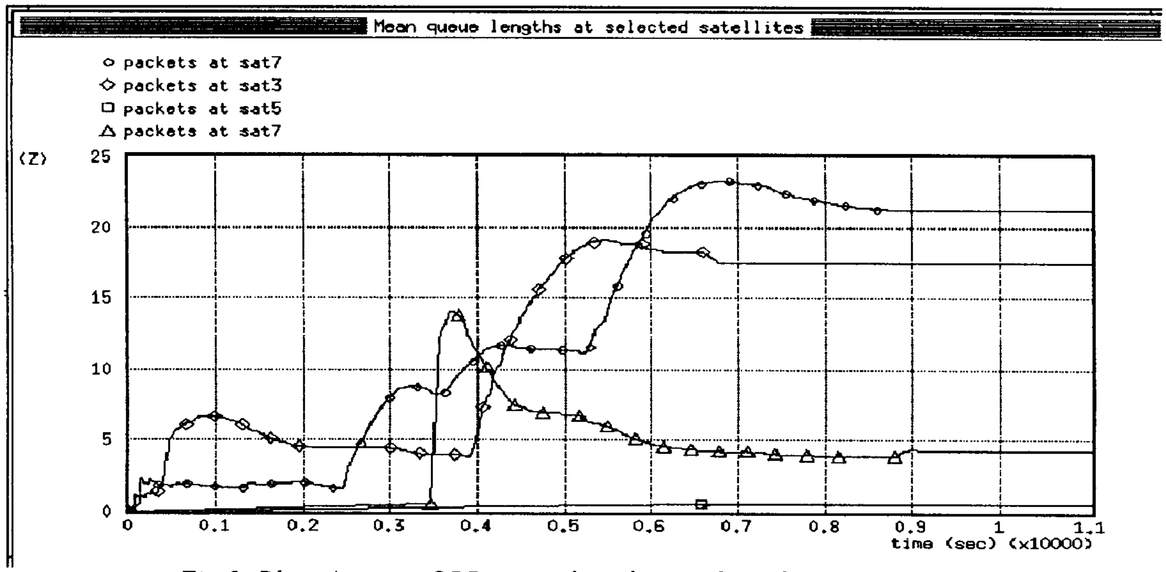



*Fig 9. Plot: Average OBP-queue lengths at selected MEO satellites*

Note that the load on each satellite in this simulation model will converge to the mean over multiple revolutions. Over a single revolution the values will not converge, as the orbital period of the ISS is not a multiple of the orbital period of the MEO network.

## 4. ATM suite in Opnet

We now describe the salient features of the ATM model as developed in Opnet 5.1. Most of the work described in the previous section deals with the ATM suite in Opnet 4.0. Additions in the newer version include process models for AAL1, 2 and AAL3/4 layers. The problems that we encountered remain the same for both versions at this stage in the model implementation. This section deals with details of the ATM model in terms of the layers that are modeled, packet formats used, ICI specifications for inter-module communication, and brief descriptions of the process models that make up the ATM suite in Opnet.

### ATM layer

The ATM layer models support many capabilities found in established as well as proposed ATM networks. Signaling, including dynamic call setup and teardown procedures, is provided for point-to-point, full duplex, Switched Virtual Circuit (SVC) service over subscription-based VPCs. The model only supports VP switching capability. Traffic control, which includes Call Admission Control (CAC) and Usage Parameter Control (UPC), prevents any calls with unsupportable traffic requirements from establishing a connection and established calls from degrading network service below specified QoS specifications. A distributed, dynamic routing capability is also provided.

a. Switching

The ATM layer supports VP and VC switching. VP switching is performed on VPCs that do not end at the switch. All switching functions are based strictly on the VPI value; the VCI values in VP switched cells are not modified. At a VPC endpoint, VC switching functions are performed, if it is not also a VCC endpoint. Delay is modeled for the translation of the incoming VPI and VCI values. Separate parameters are available for VP switching and VC switching delays, since VP switching is typically faster than VC switching. Once the output port, VPI, and VCI are determined, another parameter specifies the delay of a cell through the switch fabric. This model assumes that the VP and VC switching is sufficiently fast to support the maximum rate of arriving cells so input port buffering is not modeled. However, since it is possible to switch cells to an output port faster than the port can transmit, output port buffering is modeled.

b. Buffering

Buffers can be configured for various QoS levels at each switch. The buffer configuration applies to all ports of the switch. A QoS level is defined as the combination of the QoS category, QoS parameters and traffic parameters. The buffer configuration table associates the buffer with a specific QoS level and also specifies common buffer parameters, such as size and bandwidth allocation. An EFCI threshold may also be set for each buffer independently.

c. Call Admission

ATM traffic control functions are explicitly modeled. Call Admission Control is based on the user specified traffic contract. No call admission is applied for the UBR category. All calls belonging to the UBR category are admitted irrespective of their traffic contracts.

d. Flow Control

Usage Parameter Control (UPC) monitors cells within a particular flow and if the cells within the flow exceed the traffic contract, then actions are taken to prevent the cells from degrading the QoS of other calls. The UPC functions include determining if a cell conforms to the traffic contract as well as tagging and/or discarding non-conforming cells. The UPC function is applied only to data cells, that is, cells that do not carry ATM layer signals or routing information. The UPC function in this model is not applied to VP switched cells. This function may be optionally disabled via a parameter and is explicitly disabled if the compound cell mode is enabled.

e. Dynamic Routing

The ATM layer model supports distributed, dynamic routing. Each ATM node learns of all other connected ATM nodes and routes to them in a distributed manner, and over time, each node updates its routing tables to reflect changes in the network. Each node attempts to route a call to the destination node one hop at a time. The cost of a route is the sum of the cost of the hops. The cost of a hop is based on the amount of available bandwidth along a link. The cost can also be statically assigned by modifying the cost attribute on the links. Since there are no public standards for an ATM routing interface, a modular interface is defined that permits a transparent substitution of another routing process within the ATM layer.

### AAL layer

The AAL layer model provides AAL signaling and data transfer services. The AAL layer functionality is split into three parts: dispatch, signaling and data transfer. Each part has one or more corresponding process models, as shown in the table below:

| AAL Functionality by Model | |
|---|---|
| **Functionality** | **Supporting Process Model** |
| Dispatch function (invokes processes to handle incoming signals and data) | ams_aal_disp |
| Signaling. (call establishment and release signaling required for any AAL connection, independent of type) | ams_saal |
| Data transfer for AAL1 connections | ams_aal1_conn |
| Data transfer for AAL2 connections | ams_aal2_conn |
| Data transfer for AAL3/4 connections | ams_aal34_conn |
| Data transfer for AAL5 connections | ams_aal5_conn |

A signaling AAL (**SAAL**) connection is required to establish a new AAL connection. In this model, **SAAL** messages are sent over the ATM connection that was established to transfer the AAL PDUs. If a separate **SAAL** connection is required, the **ams_aal_disp** and **ams_saal** process models may be modified to incorporate this functionality. Segmentation and Reassembly (SAR) is modeled explicitly in the AAL models. A segmentation rate may be specified in terms of cells per second. The segmentation functionality will only generate ATM cell payloads from AAL PDUs at that rate, unless a value less than or equal to 0.0 is specified. If so, then the AAL process uses segmentation rate equal to the **PCR** of the connection. This feature implements a form of traffic shaping, which forces the AAL client's data stream to generate segments that conform to the traffic contract. As an efficiency measure, the AAL layer supports a *compound cell mode*, which prevents AAL PDUs from being segmented into smaller ATM cell payloads. Instead, the AAL and ATM layers model a compound cell, that is one composed of more than one ATM cell payload, as several virtual ATM cells. Explicit modeling of ATM cell multiplexing at switches is limited to compound cell boundaries and the UPC function is disabled. Cell Delay Variation (**CDV**) is not computed for ATM cells in this mode, since the effects of multiplexing are distorted; however, end-to-end delay statistics are collected, since the overall delay characteristics of a simulation in *compound cell mode* will be similar to the simulation in *simple cell mode*.

**Process models**

The following table summarizes the process models used by the ATM model suite in Opnet. Our simulation model only refers to a subset of this set of models.

| Name | Location | Summary |
|---|---|---|
| ams_aal1_conn | AAL module | Implements encapsulation and decapsulation of SAR_PDUs following the AAL1 protocol. |
| ams_aal2_conn | AAL Module | Implements encapsulation and decapsulation of SAR_PDUs following the AAL2 protocol. |
| ams_aal34_conn | AAL Module | Implements encapsulation and decapsulation of SAR_PDUs following the AAL3/4 protocol. |
| ams_aal5_conn | AAL Module | Implements encapsulation and decapsulation of SAR_PDUs following the AAL5 protocol. |
| ams_aal_disp | Root process of AAL module. | Creates and invokes signaling and connection processes to handle new connections, incoming signals and data. |
| ams_atm_call_dst | ATM Management Module | Handles signaling to establish and release an ATM connection at the called (destination) node. |
| ams_atm_call_net | ATM Management Module | Handles signaling to establish and release an ATM connection at a node that is neither the calling nor the called (intermediate) node. |
| ams_atm_call_src | ATM Management Module | Handles signaling to establish and release an ATM connection at the calling (source) node. |
| ams_atm_dr | ATM Management Module | Builds and updates ATM routing table; provides route recommendations to other ATM processes. |
| ams_atm_layer | ATM Layer | Encapsulates and forwards data from the |

| | | |
|---|---|---|
| | Module | AAL layer; decapsulates and forwards data to the AAL layer; forwards cells from the ATM Management module. |
| ams_atm_mgmt | ATM Management Module | Creates and invokes ATM processes to handle call signals and routing messages. |
| ams_atm_sw | ATM Switch Module | Switches cells to output ports. It also implements buffer management, based on QoS and the ABR feedback mechanism. |
| ams_atm_trans | ATM Translation Module | Receives incoming ATM cells from network;<br>Translates VPI/VCI values for outgoing cells; forwards cells to appropriate ATM module. |
| ams_ipif2 | AAL Client module | Transports IP datagrams across the ATM network; establishes and releases AAL connections, as required. |
| ams_saal | AAL Module | Handles all signaling to establish and release an AAL connection. |
| ams_traf_gen | AAL Client module | Initiates call-start and call-end. Generates the actual packets, and sends them to the AAL module. |
| ams_traf_sink | AAL Client module | Acts as the final destination for the data packets. |

**Packet formats**

The following table enumerates the packet formats used in the ATM model suite.

| Name | Description |
|---|---|
| ams_aal5_cpcs_pdu | AAL 5 PDU encapsulating an AAL 5 client SDU for transfer to the ATM layer. |
| ams_aal_signal | Carries signals within the Signaling AAL process model. |
| ams_atm_call_proceeding | Control data used to send CALL PROCEEDING message for ATM signaling protocol. |
| ams_atm_cell | ATM data cell. |

| | |
|---|---|
| ams_atm_connect | Control data used to send CONNECT message for ATM signaling protocol. |
| ams_atm_connect_ack | Control data used to send CONNECT_ACK message for ATM signaling protocol. |
| ams_atm_dr | Control cell used to implement dynamic routing within ATM. |
| ams_atm_release | Control data used to send RELEASE message for ATM signaling protocol. |
| ams_atm_release_complete | Control data used to send RELEASE message for ATM signaling protocol. |
| ams_atm_rm | Resource management cell that carries feed-back Information. |
| ams_atm_setup | Control data used to send RELEASE message for ATM signaling protocol. |

**ICI formats**

The following table enumerates the interface control information (ICI) formats used in the ATM model suite.

| Name | Description |
|---|---|
| ams_aal_handle | Contains the AAL and SAAL process handles associated with a virtual connection. Passed to ATM as an "upper layer handle" and to a client as a "lower layer handle" within ams_if_ici during call establishment and release. |
| ams_aal_request | Used within AAL to store incoming setup requests. It contains the ams_if_ici described below, as well as an interrupt code, and the object ID of the module or node sending the setup request. |
| ams_atm_handle | Passed as a "lower layer handle" within the ams_if_ici by ATM to forward state information associated with a call to AAL. It is returned to ATM during data transfer and call release. |
| ams_if_ici | Used within AMS to pass ESTABLISH and RELEASE messages between the layers of a node. It is described in further detail below. |

| | |
|---|---|
| ams_ipif_handle | Passed as an "upper layer handle" within the ams_if_ici by the ams_ipif2 process to forward connection information to AAL. |
| ams_neighbor_notify | Carries "neighbor notify" messages between the modules of an AMS node model. |
| ams_saal_ici | Used within AAL to pass signal packets to a SAAL process. |

The ATM Model Suite uses a standard ICI format to pass communications information through AAL, to and from clients at the Network Layer. This ICI is used both to open a VCC and to release one when communication has terminated. When originated by a local client, the **SETUP** and **RELEASE** requests are passed by AAL to ATM where the information is placed in control cells to be transmitted to the node at the other end of the VCC. Similarly when a control cell with a **SETUP** or **RELEASE** request from a remote node arrives at the local node, the information is passed from ATM to the local client via AAL. The format of the **ams_if_ici** is shown below.

| Attribute name | Type | Default value |
|---|---|---|
| "primitive" | Integer | -1 |
| "upper layer handle" | Structure | None |
| "lower layer handle" | Structure | None |
| "address" | Integer | -1 |
| "called party SAP" | Integer | -1 |
| "QoS class" | Integer | -1 |
| "AAL type" | Integer | -1 |
| "traffic contract" | Structure | None |
| "calling party address" | Integer | -1 |
| "AAL parameters" | Structure | None |
| "QoS parameters" | Structure | None |

The **"primitive"** is a symbolic constant identifying the nature of the signal. The **"upper layer handle"** and **"lower layer handle"** attributes are layer specific ICIs that are passed back to the originating layer on subsequent signals and data transfers. These ICIs are used by the process that created them to determine the connection that the signal or data is associated with. The **"address"** and the **"called party SAP"** together identify the client elsewhere in the network to which the signal is to be sent. The **"QoS class"**, **"AAL type"**, **"traffic contract"**, **"AAL parameters"**, and **"QoS parameters"** attributes specify information necessary for a virtual connection to be established.

## 5. Proposed ATM model

We have described the completed model implementation of the ISS-MEO network. We are presently working on replacing the proprietary MAC layer protocol with the ATM suite, so that the model can be equipped with a well-defined MAC/network layer protocol as well as AAL support for different service classes and QoS requirements. In addition, the implementation of ATM will facilitate future modeling of IP, TCP and UDP to implement end-to-end transport protocols in the simulation model.

In the following explanation of the ATM-based simulation model, the network layer shown in fig. 1 remains the same. In the future, it may be modified to increase the nodes and their connectivity, but the basic model will be retained. We have not completed the implementation of ATM in our Opnet model. This section discusses the method that we have adopted and some of the difficulties that we have encountered.

### ISS

Fig. 10 represents the node model for the ISS in the ATM implementation. The ISS traffic generator module of fig. 2 is replaced by the client module, which sits on top of the AAL module. The input-queuing module is retained, since the queuing discipline to be followed remains a test topic of interest. The beacon mechanism is also unchanged.

Among the difficulties encountered here was the sheer size of the Opnet ATM suite, with its ICI formats, neighbor notification protocol, and multiple service classes. The **`traf_gen`** module is used to create and transmit packets, and can be modified to include only a small subset of service classes. The ATM network and transmission layer, made up of the four modules **`ATM_mgmt`**, **`ATM_layer`**, **`ATM_trans`** and **`ATM_switch`**, together ensure that signaling, call setup and control, and data transmission are modeled according to the ATM standard. The ATM data packet format replaces our own packet format, and ICI setup is required to forward PDUs between modules within the ISS. State variables, simulation and node model attributes need to be included in the ISS model for proper operation of the ATM model.

The queuing mechanism is not affected by the inclusion of ATM, except in the specification of packet formats that will be seen by the queue. A separate queue and de-queuing scheme should be used for packets containing call control and signaling information.

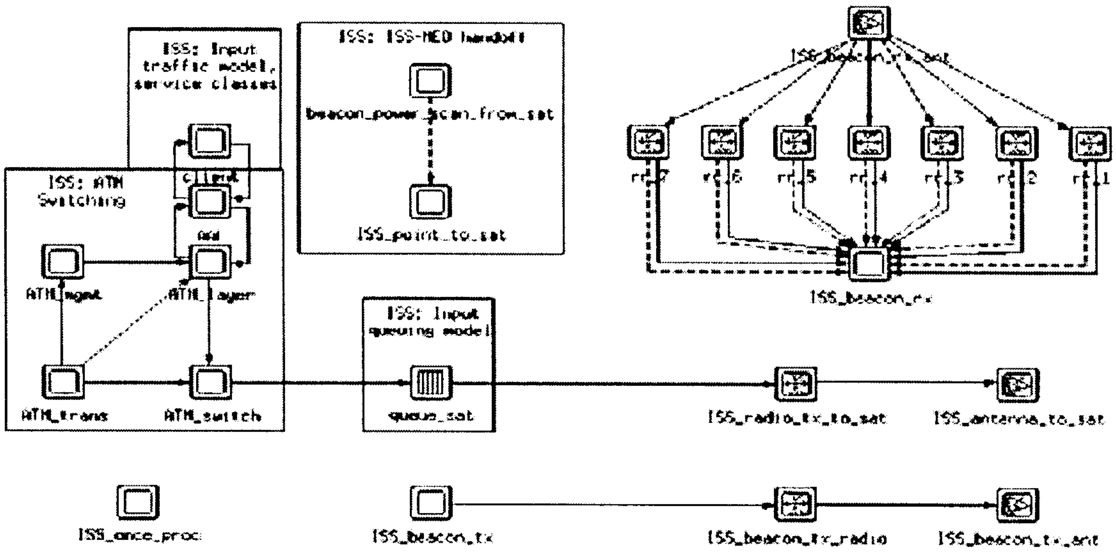



*Fig 10. Opnet node model for ISS – ATM implementation*

The beacon transmission and reception scheme need not be affected by ATM. The **ISS_beacon_rx** and **ISS-MEO handoff** modules modify model attributes that are accessed by the **queue_sat** module, so that the client module can remain unaffected.

## MEO satellite

The MEO satellite node model, shown in fig. 11, is exactly the same as the current node model shown in fig. 3, except for the ATM network layer that is added in the ATM implementation. ATM functionality is present in the Management, Translation, Switching and ATM Layer modules.

The management module performs call signaling, call setup and teardown. The ATM translation module receives incoming packets, control and data. Control packets are forwarded to the Management layer. Data packets are sent to the Switching module. The ATM layer module plays a part in call setup and in correct behavior of the satellite node as an intermediate signaling node.

Standard changes that need to be incorporated in the MEO node model include the state and temporary variable lists, node attributes, ICI formats, packet formats and simulation attributes. Modifying the **ATM_trans** module to accept packets from two queues is not a problem, and the de-queuing scheme can be implemented in the queue modules itself. The **ATM_trans** module can then be relatively undisturbed. The problem of integrating the **ATM_switch** module and the PAT subsystem is not trivial. However, a simple solution to this problem is to have an output queuing module for each transmitter that accepts packets from **ATM_switch**. This module can then make decisions based on the

PAT subsystem operations, to either send the packet or store it in a FIFO queue. In this manner, the PAT operation is completely transparent to the ATM translation, management and switching operations.

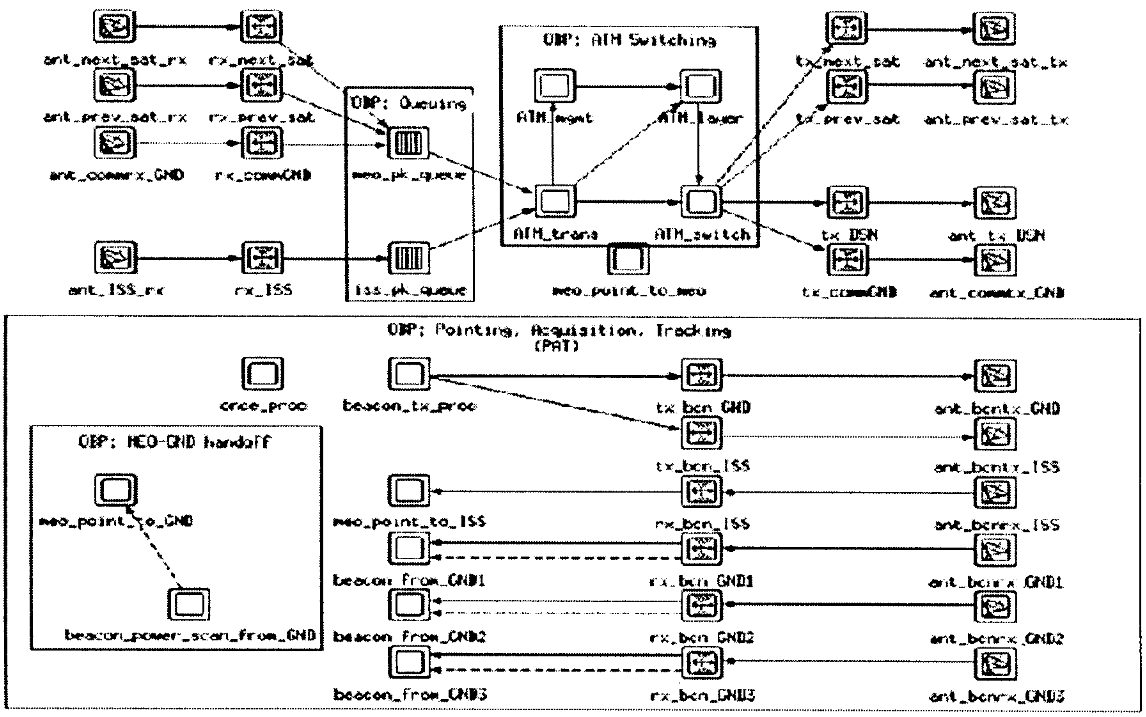



*Fig 11. Opnet node model for MEO satellite – ATM implementation*

## Ground station

The ground station also acts as an ATM switch. It receives packets from the overhead satellite and from its three directly connected network gateways, and switches the packets onto and of four outgoing paths, three point-to-point and one satellite link. In addition, the beacon tracking and monitoring operations continue as before, as shown in fig. 12.

Most of the operations that pertain to the MEO satellite node are also valid at the ground station node. We do not perform any end-to-end protocol related functions at the ground station, and so it's essential functionality is that of an ATM switch. Therefore the **commGND_to_sat** module of the earlier model (fig. 4) is replaced by the ATM switch/transmission module set comprising **ATM_trans**, **ATM_switch**, **ATM_mgmt** and **ATM_layer**. The **ATM_trans** module is modified to accept packets from two incoming queues, with the de-queuing scheme implemented at the queue modules.

For easy integration of the ATM functionality with the beacon tracking system, an output queue at each transmitter can be used to ensure that the **ATM_switch** module does not have to be modified to access beacon signal arrays and make pointing and transmission decisions. The queue modules can then communicate using shared variables with the PAT subsystem to determine if a packet can be transmitted or must be queued.

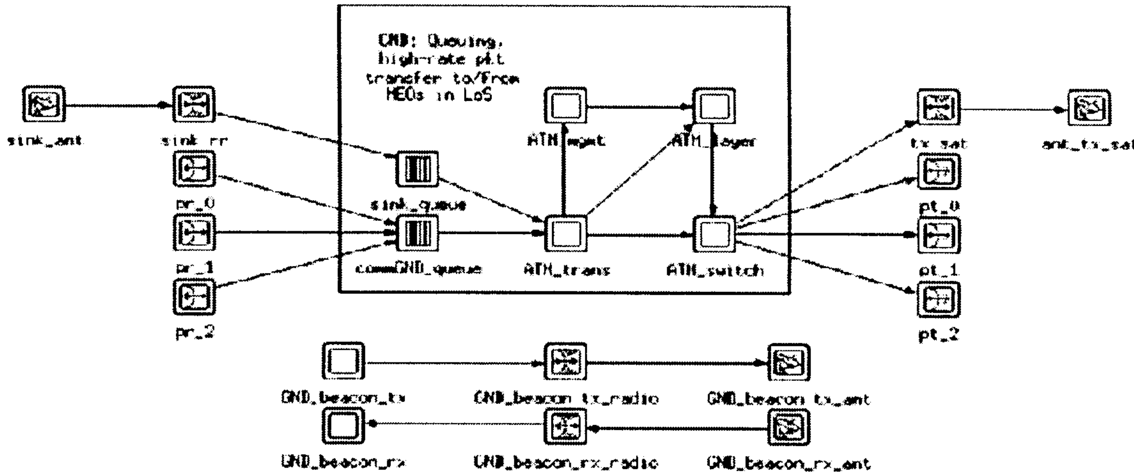



*Fig 12. Opnet node model for GND station – ATM implementation*

The input queuing modules from the current GND station implementation in fig. 4 are retained here, to separate the problem of ATM switching from that of queuing and priority management. The beacon tracking modules are unchanged. State and temporary variables, model attributes and simulation attributes need to be added. Both MEO satellite nodes and the ground gateways implement the `call_net` process model of **ATM_mgmt**, while the network gateways and ISS implement `call_src` and `call_dst` models.

**Ground terminal/Network gateway**

The network gateway node is modeled as an ATM end client, like the ISS. Therefore, the entire ATM and AAL stack is implemented in the node as shown in fig. 13. The client sits on top of the AAL stack and can be modeled independently of it. In the present plan, the queuing module is separate from both ATM and client modules, and performs operations like SAR, duplicate detection and packet sequencing.

The **traf_sink** module is used as a client module, with modifications to allow for the collection of end-to-end statistics such as packet delay and loss, and file transmit times. The network gateways also act as sources of traffic, and so the **traf_gen** module is also used to generate packets that are destined for the ISS or for other network gateways. Since the network gateways and the ISS are end nodes in this simulation model, they carry the entire protocol stack on them. This may include, in later versions, IP support for multiple datagram services with varying QoS, and a transport protocol like TCP or UDP for end-to-end protocol operation and simulation of Internet traffic.

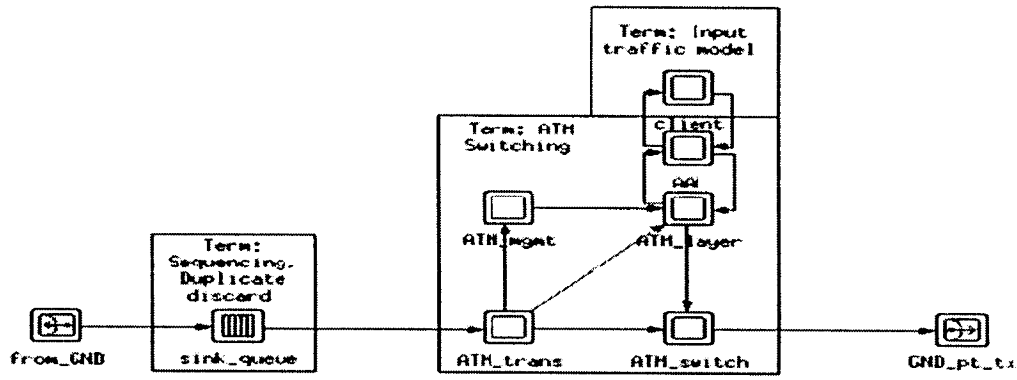
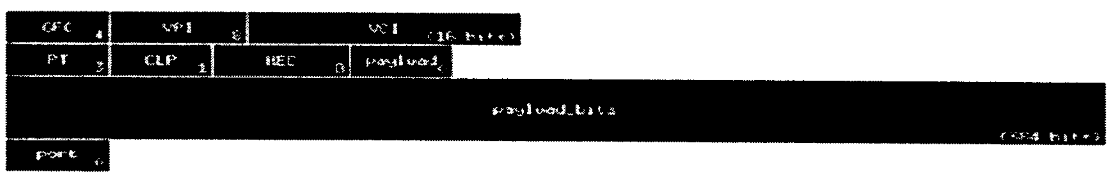



*Fig 13. Opnet node model for ground terminal/network gateway – ATM implementation*

These operations, however, can be merged with the ATM functionality, or can be performed at the client module. The receiver queue module in network gateway node does not perform any special functions and is not a part of the protocol test cases.

**Packet formats**

The ATM suite in Opnet is comprised of multiple packet formats, ranging from AAL1 and AAL2 SAR PDUs to the ATM cell format and various ATM call control, setup and signaling packets. For example, fig. 14 shows two packet formats that will be used in the ATM implementation – the ATM cell and the AAL3/4 SAR PDU.




*a)*


*b)*

*Fig 14. Packet format for a) ATM cell, b) AAL3/4 SAR PDU*

## 6. Further additions

Some additions have been made to the original model described in section 3. Essentially a very basic performance comparison is implemented between the 7-MEO satellite network model and a 3-GEO network model. The 3-GEO model is briefly described here.

### Network model

Figure 15 shows the network model for the 3-GEO case. The only changes are in the number and positions of the satellites. The orbit files used to generate the orbits of GEO satellites are derived from standard examples of orbit files in Opnet. The satellites are placed 120 degrees apart in longitude. Each GEO satellite is in continuous visibility of one GND gateway node. The antenna patterns for transmit antennas on the GEO satellite nodes have a very sharp cone. However, because of the altitude of the satellites, there is considerable overlap of satellite footprint. This leads to the generation of many duplicate packets in the network.

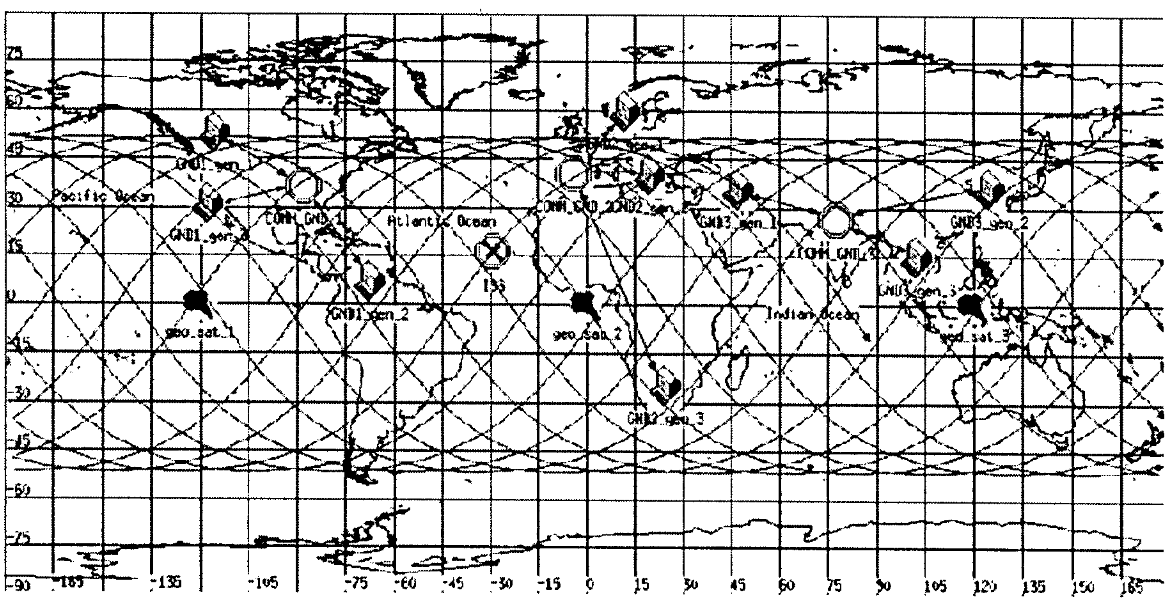



*Fig 15. Opnet network model for ISS and 3-GEO constellation*

### Node models

We have developed the different components of the simulation model in a modular, object-oriented manner. Therefore, the only changes that have to be made in converting from 7 MEO satellites to 3 GEO satellites is to replace the component that is used to model the satellite node. The PAT and handoff modules of **meo_node** are simplified in

**geo_node**. All other operations of the GEO node, include **geo_proc**, **geo_pk_queue** and **iss_pk_queue** are unchanged.

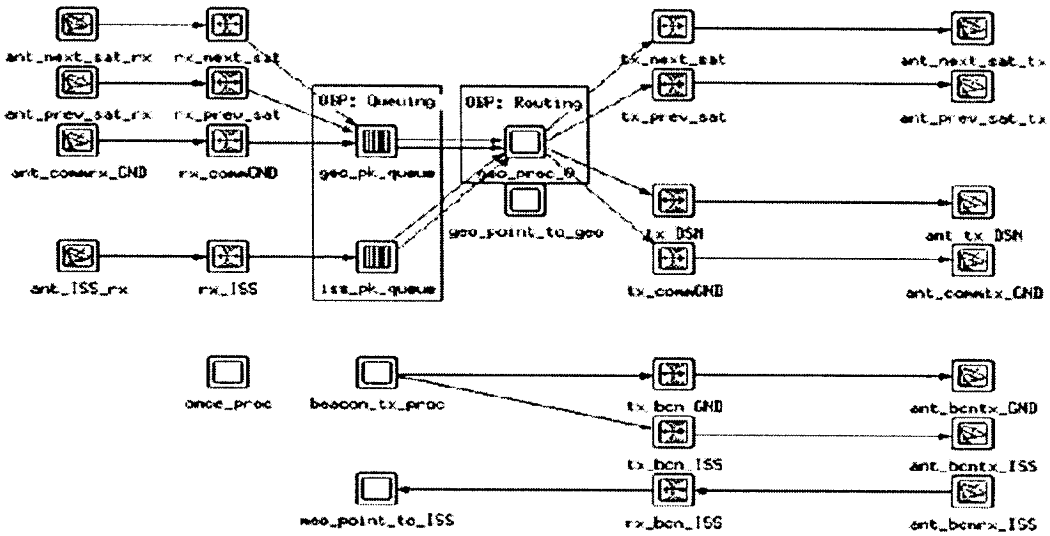



*Fig 16. Opnet node model for GEO satellite*

There is no use for beacon tracking between ground gateways and geostationary satellites. As shown in figure 16, the PAT subsystem is much simpler in the GEO case. Modules like **tx_bcn_GND** and **tx_bcn_DSN** are not used. The **tx_bcn_DSN** transmit module can be used in future to interface to NASA's TDRSS-DSN network. Transmitter power is higher in the GEO case, between satellites and from satellites to ground gateways. Receiver gains are not changed.

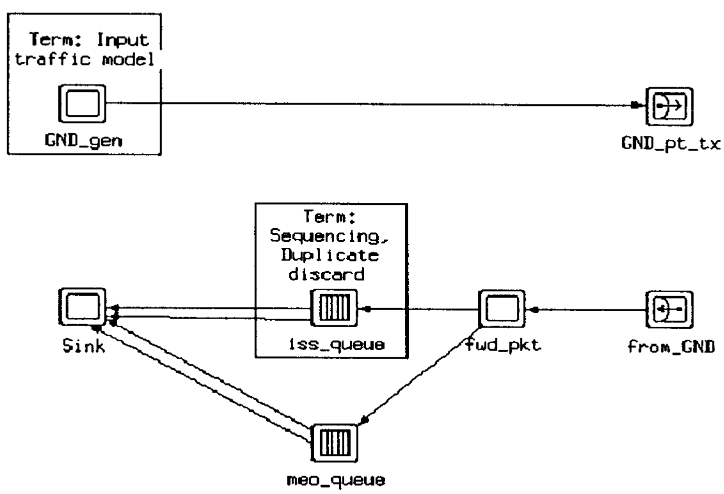



*Fig 17. Opnet node model for ground terminal/network gateway*

To facilitate the collection of end-to-end delay statistics, a global statistic variable is used. All packets are time-stamped at the time of creation and the destination network gateways record the time that each packet took to reach. This transfer time per packet is recorded in the global statistic array by all the destination nodes. We only record packet delays for packets that originate at the ISS. Packets that are generated at other network gateways are treated as background traffic, and they do not have any relevance to measurement. To allow for this, the network gateway node model is modified by the inclusion of separate receive queues for ISS traffic and background traffic. This allows us to isolate and study the effect of protocol operation and queuing schemes at the receivers (end-user nodes in the Internet) on bandwidth-intensive multimedia applications. Operations like SAR, duplicate detection and packet sequencing are only performed for ISS traffic.

## 7. Conclusion

We are developing a large-scale simulation model to evaluate the feasibility of carrying NASA mission payload, command and control, real-time and low-priority data between ground user terminals and near-earth spacecraft, using proposed commercial satellite constellations. The simulation model will allow us to perform detailed studies to quantify the performance of satellite systems for the following test parameters: specific services and their QoS requirements, protocols, traffic models, satellite routing schemes, on-board bandwidth/buffer allocation methods, queuing disciplines, and handoff strategies.

In this report, we have explained some of the features of the present Opnet model. The components of the model follow the requirements set down by the proposal that was submitted in January '99. A quasi-MAC protocol has been implemented over the entire network, and fundamental test cases have been identified. Test modules will be developed independently in Opnet to simulate the operation of each test case, including input traffic models, service classes, queuing disciplines, bandwidth assignment algorithms, routing algorithms, coverage issues and handoff schemes. End-to-end performance will be quantified and analyzed.

Moving this model to ATM is not a trivial task. Much work has been done in attempting to understand the intricacies of the Opnet ATM model. Section 4 in this report is a short summary of that work. The real problems come in understanding and working with some of the complicated structures, ICIs and lists that are used in the ATM models in the operation of neighbor notification, dynamic routing, addressing, call setup and call management. Not all of it has been covered so far. Most of our work has been limited to the working of the four ATM switching and transmission modules (**`ATM_mgmt`**, **`ATM_layer`**, **`ATM_switch`** and **`ATM_trans`**) and their process models.

Section 5 then deals with the more important task of integrating the ATM model with our current model. Much of our current functionality will be replaced, like that of packet generation, routing and receiver operations (SAR, duplicates, sequencing). Wherever possible, we are attempting to insulate the ATM modules from our own, so that they can retain much of their identity. This necessitates the creation of additional modules that will interface between the two sub-models. An example of this is the output queue modules that are proposed at the MEO nodes and the ground gateways, so that the ATM switching and transmission functions can be isolated from the PAT, beacon tracking and LoS-related functions.

It is also worth stressing that implementation of ATM is not the final goal here. It is advisable to first consider whether it is even necessary to implement ATM in the simulation model. The goal of the simulation, as explained earlier, is to create a baseline model and then create plug-ins of test cases to try and evaluate the problem of Internet support for the ISS and other NASA missions. The complicated nature of the ATM simulation model is itself a strong argument against it. We can, instead, consider implementing a simple, proprietary MAC transmission and switching protocol and then working on implementing IP and TCP/UDP support over this MAC layer. By replacing the MAC layer with ATM, we originally sought to make this problem well-defined and bring it into the mainstream, since the IP-over-ATM problem and the ATM-over-LEO-satellite problem are both very popular and have generated a lot of research interest. However, keeping in mind the goal of our research effort, we may be better served by abandoning the complications of implementing full-scale ATM. We can look instead at ways to implement TCP/IP or a variant over a proprietary transmission/switching protocol that is specific to the needs of the space segment. In that case, section 5 can be omitted and we can continue with our work on the earlier version of the model.