Space Communications and Navigation (SCaN) Network Simulation Tool Development and Its Use Cases

Esther Jennings[1], Richard Borgen[2], Sam Nguyen[3], John Segui[4],
Tudor Stoenescu[5], Shin-Ywan Wang[6], and Simon Woo[7]
Jet Propulsion Laboratory/California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109, USA

Brian Barritt[8], Christine Chevalier[9] and Wesley Eddy[10]
NASA Glenn Research Center, 21000 Brookpark Road, Cleveland, OH, 44135, USA

In this work, we focus on the development of a simulation tool to assist in analysis of current and future (proposed) network architectures for NASA. Specifically, the Space Communications and Navigation (SCaN) Network is being architected as an integrated set of new assets and a federation of upgraded legacy systems. The SCaN architecture for the initial missions for returning humans to the moon and beyond will include the Space Network (SN) and the Near-Earth Network (NEN). In addition to SCaN, the initial mission scenario involves a Crew Exploration Vehicle (CEV), the International Space Station (ISS) and NASA Integrated Services Network (NISN). We call the tool being developed the SCaN Network Integration and Engineering (SCaN NI&E) Simulator. The intended uses of such a simulator are: (1) to characterize performance of particular protocols and configurations in mission planning phases; (2) to optimize system configurations by testing a larger parameter space than may be feasible in either production networks or an emulated environment; (3) to test solutions in order to find issues/risks before committing more significant resources needed to produce real hardware or flight software systems. We describe two use cases of the tool: (1) standalone simulation of CEV to ISS baseline scenario to determine network performance, (2) participation in Distributed Simulation Integration Laboratory (DSIL) tests to perform function testing and verify interface and interoperability of geographically dispersed simulations/emulations.

Nomenclature

AOS = Advanced Orbiting System
AR4JA = Accumulate Repeat by 4 Jagged Accumulate
CCSDS = Consultative Committee for Space Data System
CEV = Crew Exploration Vehicle
CLV = Crew Launch Vehicle
CxP = Constellation Program
DSIL = Distributed Simulation Integration Laboratory
ENCAPS = Encapsulation Service

[1] Communication Networks Architectures and Research Engineer, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[2] Senior System Engineer, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[3] Network Analysis Engineer, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[4] Telecommunication System Engineer, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[5] Telecommunication Networks Architect, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[6] Senior Communication Networks Engineer, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[7] Telecommunications System Engineer, NASA JPL, 4800 Oak Grove Drive, Pasadena, CA 91109.
[8] Space Communications Network Architect, DB Consulting Group / NASA GRC, 21000 Brookpark Road, Cleveland, OH 44135.
[9] Electrical Engineer, Analex Corporation / NASA GRC, 21000 Brookpark Road, Cleveland, OH 44135.
[10] Space Communications Network Architect, Verizon Federal Network Systems / NASA GRC, 21000 Brookpark Road, Cleveland, OH 44135.

GN = Ground Network
IMSim = Integrated Mission Simulation
Kbps = Kilo bits per second
M_PDU = Multiplexing Protocol Data Unit
MLB = Master Link Book
NEN = Near Earth Network
NISN = NASA Integrated Services Network
SCaN NI&E = Space Communications and Navigation Network Integration and Engineering
SN = Space Network
STK = Satellite Took Kit (software by Analytical Graphics, Inc.)
TOAST = Telecom/Orbital Analysis Tool
TDRSS = Tracking and Data Relay Satellite System

I. Introduction

One of the NASA's visions is to continue the exploration of space, specifically, to return humans to the moon and continue on to Mars. As space-based networking differs from terrestrial networking, interplanetary communication protocols need to be designed, validated and evaluated carefully to support different mission requirements. As actual systems are expensive to build, it is essential to have a low-cost method to validate and verify mission/system designs and operations. This can be accomplished through simulation.

Simulation can aid design decisions where alternative solutions are being considered, support trade-studies and enable fast study of what-if scenarios. It can be used to identify risks, verify system performance against requirements, and as an initial test environment as one moves towards emulation and actual hardware implementation of the systems.

Models for non-standard, non-COTS protocols used aboard space systems are not readily available in commercial software[1]; thus making it difficult to simulate the envisioned exploration network. The core of our simulation tool is the commercially available QualNet network simulator. Prior to this work[2], we had investigated IP-based networking protocols for space exploration and simulated a 14-day mission using shuttle data. The objectives of the previous work were to determine whether SCaN can meet the communications needs of the mission, to demonstrate the benefit of using QoS prioritization, and to evaluate network key parameters of interest such as delay and throughput. However, since then we have improved the fidelity of the simulator by adding customized space protocol models and physical layer models; previously, physical layer characteristics were read in from the JPL developed telecom/orbital analysis tool (TOAST).

In the past two years, we focused on developing a simulation tool (SCaN NI&E Simulator) to assist in analysis of current and future (proposed) network architectures for NASA. Specifically, the Space Communications and Navigation (SCaN) Network is being architected as an integrated set of new assets and a federation of upgraded legacy systems. The SCaN architecture currently includes the Space Network (SN), the Near-Earth Network (NEN) and the Deep Space Network (DSN). The space segment of the SN element consists of multiple operational Tracking and Data Relay Satellites (TDRS) in geosynchronous orbit at allocated longitudes. The TDRS System (TDRSS) relays forward and return service signals to and from customer spacecraft, thereby providing data transfer and tracking services. The GN sites primarily support S-band communication links. The White Sands Complex (WSC) provides the communications equipment necessary for transmitting and receiving data and tracking information relayed via each TDRS. The NASA Integrated Services Network (NISN) provides wide area network (WAN) telecommunications services for the transmission of terrestrial data, voice, and video between all SCaN Network ground elements and Constellation/user ground elements. Future Lunar missions may involve the Deep Space Network (DSN) and Lunar Relay Satellite(s).

The baseline scenarios for stand-alone and DSIL tests involve the following elements: the Crew Exploration Vehicle (CEV), the Crew Launch Vehicle (CLV), the International Space Station (ISS), three Tracking and Data Relay Satellites, the White Sand Complex (WSC), TEL-4 (Air Force Ground Station) and Constellation's Mission Control Center (MCC). There are five main phases of the mission scenario: countdown, ascent, low Earth orbit, rendezvous with ISS, and descent (return to Earth). Each mission phase may be further divided into sub-phases. For example, rendezvous with ISS has three sub-phases (far, near and very near) depending on the distance of the CEV to ISS. During each of these mission phases, the data traffics are of different types, different distributions and different volumes. Experiments were run to determine whether SCaN's configuration can support the nominal data traffic efficiently. Performance metrics of interests are: delay, throughput, and jitter.

The protocol stack proposed for the CEV-ISS mission uses the IP-protocol as well as Consultative Committee

for Space Data Systems (CCSDS) protocols. However, off-the-shelf QualNet models do not include CCSDS protocols. Thus, we developed the following custom models and linked them with QualNet: (1) Link Budget Library based on information contained in the Master Link Book for the CEV-ISS mission, to model physical layer behaviors; (2) SCaN Physical Model that includes TDRSS loss of signal and acquisition of signal model and an interface to the Link Budget Library and (3) CCSDS AOS data link layer Multiplexing Protocol Data Unit Model; (4) CCSDS Encapsulation Service Model; and (5) SCaN Network Traffic Model. Future work will incorporate SN models in the Satellite Took Kit (STK) to be integrated into the physical layer model of the simulator.

We developed an effective experiment management system for the definition and batch simulation of Constellation scenarios using the SCaN NI&E Simulator. The team has employed this system to simulate most of the space link scenarios defined in the Constellation Computing System Architecture Design Document (CSADD)[3], enabling improvements in the simulation models and validating the design assumptions of the Constellation communications scenarios. We ran experiments for each mission phase and its sub-phases. Traffic types include representative telemetry, commands and voice. Each traffic type has an associated forwarding priority scheme. We controlled the data volume to fit within the network's bandwidth capacity (nominal traffic) and observed that all application-layer data would be delivered without loss, as expected. We generated representative voice traffic, and observed that jitter is less than 30ms in most cases where 40ms is regarded as a threshold to guarantee voice quality[10]. Higher jitter is observed during countdown from MCC to CEV (having a jitter of 50 ms, 192kbps bandwidth) and during ascent from MCC to CEV (with a jitter of 64 ms, 72 kbps bandwidth). The constrained bandwidth may have caused the higher jitter value.

The SCaN NI&E Simulator has been used to support DSIL testing. The goal of DSIL is to support integrated testing as early as possible, to test performance and do early risk mitigation. DSIL experiments contained geographically distributed simulation components (CEV, CLV, SCaN, ISS, MCC) interconnected through the High Level Architecture (HLA), which is a general-purpose architecture for distributed computer simulation systems (HLA is defined under IEEE standard 1516). The testlabs are connected through NISN. Specifically, the SCaN NI&E Simulator has also been used to perform delay trade studies, and it has further been used in DSIL experiments for relaying data, imposing delay, and verifying connectivity to TDRS and WSC. The simulator computes the bit /frame error rate (BER/FER) according to the Master Link Book information[4]. In nominal test cases, the BER/FER is very low that we do not see the data being dropped, which is what we expected.

As more information becomes available concerning the spacecraft and the mission, we will update our custom models to provide higher fidelity simulations. After testing the nominal mission scenarios extensively, we are adding off-nominal test cases to determine the network's capability to handle or recover from faults.

In Section 2, we describe the simulator design. Custom models are presented in Section 3, followed by the simulator's use cases in Section 4. Conclusion and future work are presented in Section 5.

I. Simulator Design

At the core of the SCaN NI&E Simulator is the QualNet engine, a commercial product from Scalable Network Technologies (SNT). The architecture of the QualNet simulation engine closely resembles the five layers of the TCP/IP model (where the session and presentation layers of the OSI model are omitted). The underlying QualNet simulator includes over one-hundred network protocol and radio propagation models off-the-shelf. However, in its off-the-shelf configuration, QualNet does not include certain models required for simulation of SCaN and/or Constellation Program (CxP) space communications. To solve this problem, the SCaN NI&E Simulator team extended QualNet by adding custom protocol models. The custom protocol models were developed in C++ using the QualNet API for its model and module libraries. In Figure 1, we show the custom SCaN NI&E Simulator models and where they belong in the TCP/IP reference model. Each model is implemented at one of the five layers, which communicates with one another (up or down the communications stack) using standard API calls.

During execution of a simulation, the simulator creates an instance of the five module layers and their associated models for each node in the simulation, as specified by the user. To simulate communication between two nodes at the higher layers, the format of the data is modified appropriately by the custom models as packet overhead is added and the payload is disassembled into frames. Delays for error-checking and retransmission are also simulated. Once the data is ready to be transmitted by a node's physical layer, the underlying discrete-event simulation engine schedules an event to occur at the receiving node after accounting for appropriate transmission and propagation delays. Frames are randomly dropped by the receiving node according to the calculated bit-error-rate (BER) and/or frame-error-rate (FER) of the transmission across the modeled propagation medium. Abstract data then travels up

the communication stack of the receiving node for packet reassembly and processing by the application. Statistics are gathered throughout this process for later analysis by the user.

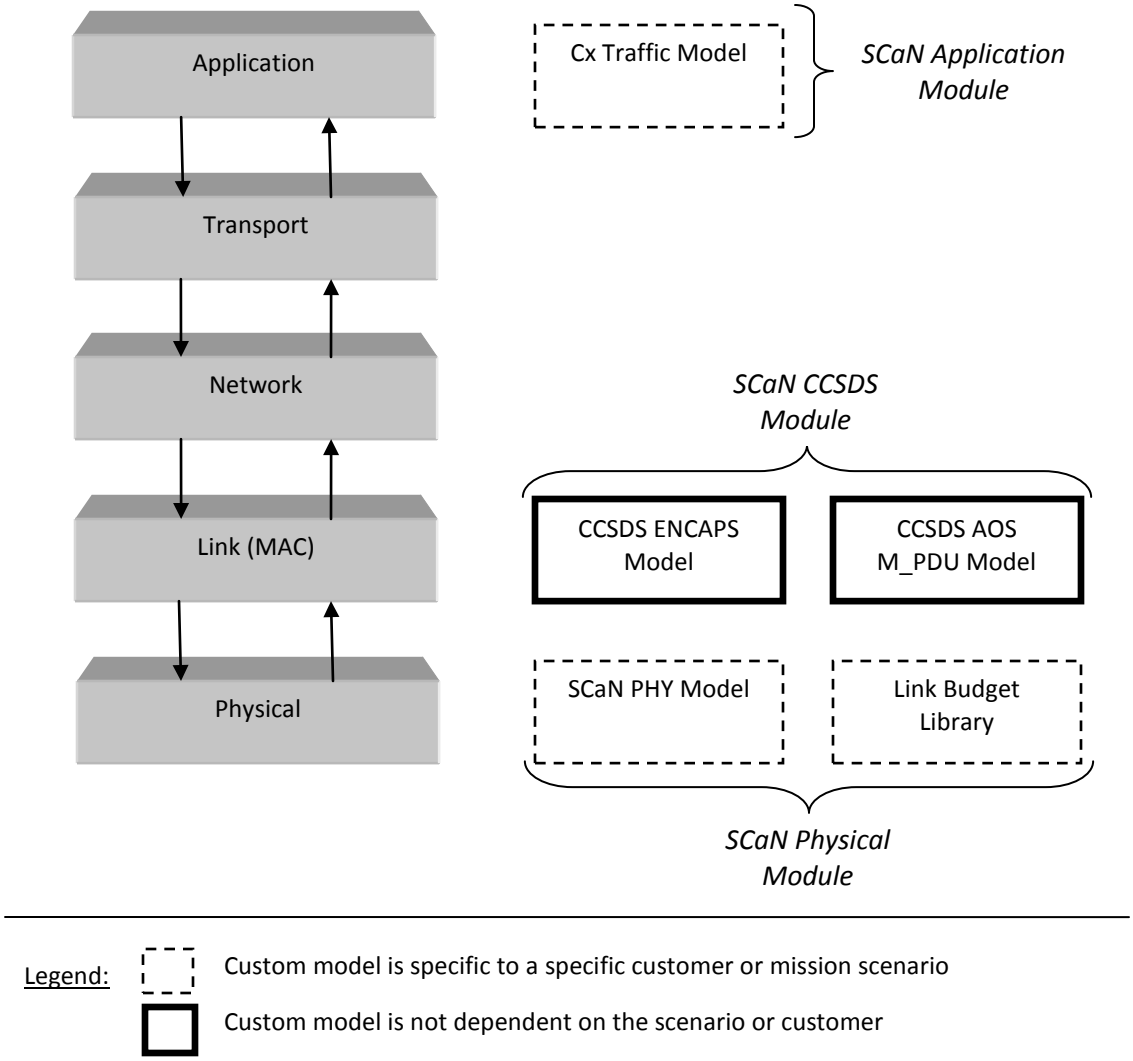

Figure 1 – Custom models developed and contributed to the QualNet model library

II. Custom Models

While the QualNet network simulation tool comes with a wide range of protocol models, it does not have models for space-based networks that use CCSDS protocols. The SCaN NI&E Simulator focus on the development of the following custom models: (1) Link Budget Library, (2) SCaN Physical Model, (3) CCSDS AOS Multiple Protocol Data Unit Model, (4) CCSDS Encapsulation Service Model and (5) SCaN Network Traffic Model.

A. Link Budget Library

The Link Budget Library includes link performance analysis subroutines for the following links at different phases and different bands (S-, Ka-bands): CEV to SN (TDRSS), CLV to SN (TDRSS), and CEV to ISS. The library subroutines are written in the C++ programming language and can be executed on both Windows and Linux platforms. Values of the configurable communications parameters are set according to information extracted from the CxP Master Link Book[4]. Dynamic parameters (e.g. spacecraft positions) are provided when calling the subroutines from an external program; in this case, the customized SCaN Physical Model (described in the next subsection). Using the spacecraft positions, the link budget library routine computes the dynamic space loss

(indicated as red text in Fig. 2). The link budget library provides calculations of bit error rate and frame error rate. These are used by the SCaN Physical Model to statistically model bit errors and frame errors on the physical channel. In future versions of the link budget library, we plan to incorporate antenna patterns into the link budget calculations.

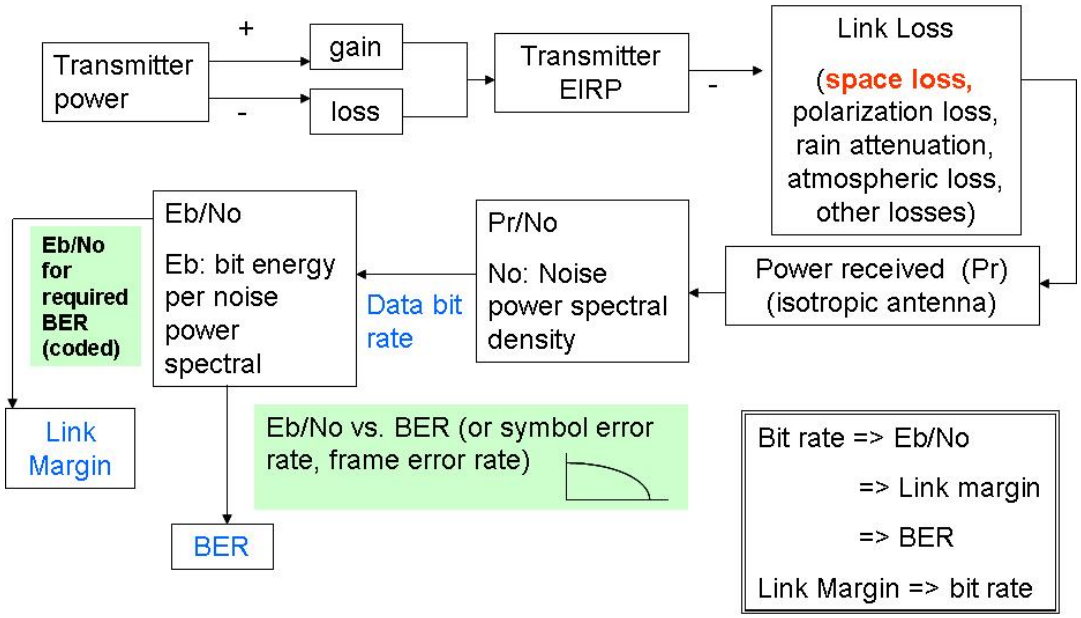

Figure 2 – Link Budget Calculation

B. SCaN Physical Model

There are two components in the SCaN Physical model: (i) TDRSS models and (ii) the physical propagation model. TDRSS models include the TDRSS handover model and bent-pipe model. The physical propagation model is integrated with the Link Budget Library to accurately compute the link budget between any two spacecrafts based on their positions, frequency, noise, power, antenna gains, etc. For TDRSS selection/handover, the model uses periodic distance checks. We assume a handover is scheduled when CEV crosses mid-point between two TDRS satellites. The TDRS selection criteria model is based on CEV-TDRS range and line-of-sight. Users may specify the time required for acquisition of signal, which may include time needed for antenna slew, carrier acquisition, etc. TDRSS switching is modeled by using outbound queues to switch between TDRS satellites. If a TDRS handover is scheduled in the middle of the transmission of a frame (which should not happen), the frame will finish the transmission to the same TDRS (without switching); in other words, we do not send part of a frame to one satellite and the other part to another satellite. If there is an outage interval during the handover, frames will be dropped.

The physical propagation model accepts node trajectory as input and passes the information to the link budget library for BER calculations. Transmitter and receiver parameters are configured in the Link Budget Library depending on the mission phase. Frequency/band change (from S to Ku) is modeled in the link budget library. Currently, we do not model cross-channel interference. TDRS model uses a stochastic model to drop frames according to BER and use distance to determine propagation delay. There is no buffering at TDRS; it is a bent-pipe.

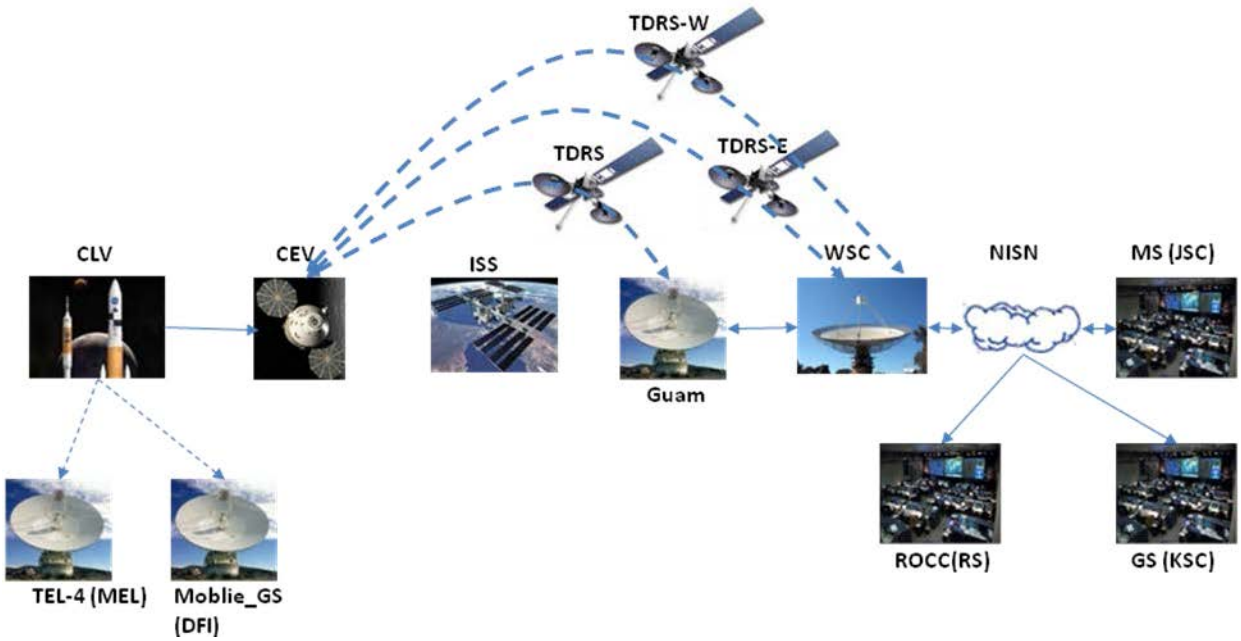

Figure 3 – Topology showing SCaN Physical Model with TDRS bent-pipe to WSC / Guam

C. CCSDS AOS Multiplexing Protocol Data Unit Model

The AOS Multiplexing Protocol Data Units (M_PDU) model is implemented in the MAC layer within the simulator. The AOS M_PDU model accurately updates the size of a packet to account for AOS transfer frame headers, channel coding (with configurable rate), and frame synchronization markers. The extra bytes that are added to packets by ENCAPS are also accounted for when propagating the frames over the physical layer models, when ENCAPS is enabled on an interface. The CCSDS AOS M_PDU protocol is specified in the CCSDS 732.0-B-2 document, a "blue book" recommendation[5]. The AOS M_PDU service has several components that contribute to an increase in the number of transmitted bytes versus the size of a packet at the network layer:

- AOS Transfer Frame Primary Header – mandatory - 6 to 8 bytes
- Transfer Frame Insert Zone - optional – not implemented in the current version of the SCaN NI&E Simulator
- Transfer Frame Trailer – optional – 6 bytes
- M_PDU Header – mandatory – 2 bytes
- M_PDU Packet Zone – includes ENCAPS packets
- Fill bits needed to make the Transfer Frame the correct length for the channel coding
- CCSDS Channel Coding – variable coding rate
- Frame Synchronization Markers – variable length, depends on channel coding

The M_PDU Packet Zone can include multiple packets. The first and last packets in the Packet Zone may or may not be complete. Figure 4 shows AOS ENCAPS, Transfer Frame, M_PDU, and LDPC Overheads[6].

Additionally, there is some computational latency involved in framing and coding packets in preparation for being propagated, but only the coding delay is likely to be significant in comparison to the propagation delay of space links. For Constellation, the AR4JA LDPC code has been selected, at rate ½ and with a 1024 bit information length. The latency implied by this code is about 14 ms according to available estimates[7] and this is incorporated into the SCaN Physical model.

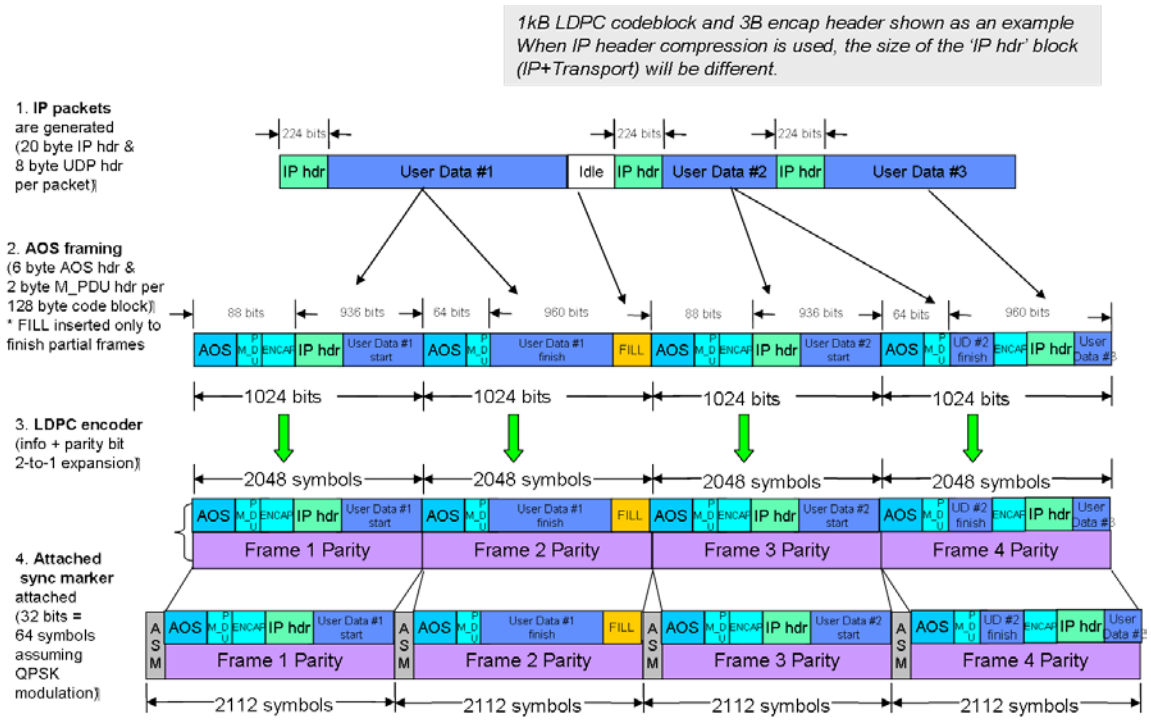

Figure 4 - AOS ENCAPS, Transfer Frame, M_PDU, and LDPC Overheads

D. CCSDS Encapsulation Service Model

The CCSDS ENCAPS protocol is specified in the CCSDS 133.1-B-1 document, a "blue book" recommendation[8]. Since CCSDS data link protocols do not natively support the IPv4 or IPv6 network protocols, various techniques have been developed in order to transport IP packets over CCSDS data links. One of these techniques is the use of the ENCAPS protocol within the AOS M_PDUs. This is the technique that has been agreed upon between SCaN and Constellation for supporting the CEV-ISS mission.

A simple explanation of this use of the ENCAPS service is that it makes up for the fact that AOS lacks a field equivalent to the "ether-type" field that terrestrial protocols typically use (e.g. in 802.2 LLC) to indicate the upper layer protocol that is being transferred. An ENCAPS header provides a 3 bit "Protocol ID" field for identifying upper layer protocols, with one of the 3-bit code-points defined to mean that another optional 4-bit field "Protocol ID Extension" field is also used. An ENCAPS header also includes a variable length (0- to 4-byte) field that encodes the length of the encapsulated upper layer packet. Using these type bits, AOS or other CCSDS data links can use ENCAPS as one way to support upper layer protocols that are not explicitly supported in the data link's design.

Use of ENCAPS adds between 1 and 8 bytes to each upper layer packet, before framing, coding, and transmission across the data link (removing those bytes on reception). With small payloads, and low rate links, this may have an impact that affects simulation results. The work of encapsulation and de-capsulation performed by ENCAPS is quite simple computationally and should not result in an amount of delay that has impact on a simulation, as it will be many orders of magnitude less than the propagation delay of space links.

E. SCaN Network Traffic Model

The traffic models supported by QualNet do not provide accurate models for simulating space network traffic. We developed custom traffic models that capture the fundamental traffic types that will be found in the space communications environment. This model provide the capability to investigate the new IP-based communications functions and resulting performance that are required by the human exploration missions of NASA's Constellation Program and the Space Communications and Navigation (SCaN) network's ability to support them.

Four prototypical traffic models have been developed, from which any number of streams may be generated: (1) audio/voice, (2) real-time motion imagery (live video), (3) telemetry (including both regular periodic data as well as intermittent file downloads), and (4) command (including periodic data as well as intermittent subroutine uploads).

The voice traffic model stochastically parameterizes calls, talk times, and talk spurts. Multiple audio analysis tools were developed to assist in the derivation of parameters using data samples taken from relevant space missions. Using these tools, parameters of probability distributions were fit to data taken from Shuttle/ISS archives.

The characterization of motion imagery (video) traffic involved the stochastic characterization of I-, P-, B-frames of MPEG-4 sequences. Several Matlab tools were developed to support video analyses, wherein ISS and Shuttle specific mission video sequences were used to derive parameters for the associated model.

A general stochastic model was derived for data telemetry enabling broad parametric applicability, ranging among constant bit streams, periodic, or bursty patterns. This same general model, through proper selection of parameters, may also be used to model command data flow traffic.

III. Simulator Use Cases

The SCaN NI&E Simulator supports two modes of operation: (1) Stand-Alone Simulation and (2) Integrated DSIL Operation (distributed simulation).

A. Stand-Alone Simulation

One of the most powerful uses of simulation is to exercise a large number of scenarios in a rapid and controlled fashion. This can be the most efficient means to support trades, sensitivity analysis, regression testing and so forth. The SCaN/DSIL Simulation/Emulation Team has developed an effective experiment management system for the definition and batch simulation of Constellation scenarios. Batch mode simulation makes it possible to run many tests automatically, repeatedly and much faster than real time. This system has been employed to simulate most of the space link scenarios defined in the Constellation Computing System Architecture Design Document (CSADD), enabling improvements in the simulation models and validating the design assumptions of the Constellation communications scenarios.

The experiment management system defines external simulation requests and their final simulation results in neutral formats where these neutral formats are cast in terminology and references that conform to the Constellation system and the Constellation design documents. Specifically, the request and results formats are XML documents defined and validated by XML schemas. The XML documents support summary reporting, long-term archival of simulation runs and thorough documentation of the assumptions, design, software versions and other general contexts of each simulation. The experiment management system reads and interprets the simulation request in order to configure the SCaN NI&E Simulator scenario, execute the simulation, and finally gather and reformat the results.

From the CSADD document, we extracted 28 basic experiments from launch/ascent to return/descent. There is an XML file associated with each experiment; in the XML file, the link identity, data rate, traffic type and characteristics are specified. An XML schema (.ssd) file is used to check the legal syntax, tag names and permitted value types. From the CSADD, we generated XML files for the experiments. After these XML files are checked using the XML schema, the experiments are run in batch mode by invoking a Perl script. Each experiment has an associated directory where the configuration files and statistic files reside. This directory structure ensures configuration control of the experiments and enables repeatable runs with reproducible results. After the batch job completes, additional Perl scripts are invoked to filter relevant statistics and generate a summary report. Figure 5 provides an example of a report summarizing multiple simulation runs.

Sim	Test Name	Sim Date	CSADD Needline	Pkts Sent	Thruput (bps)	Latency	Jitter	Pkt Loss
1	CLOSE RENDEZVOUS CEV-A-ISS – Baseline	2008-12-24	CEV-ICCA-TLM-docked-MOD	4200	16003	90	18665	0
2	FAR RENDEZVOUS CEV-A-ISS – Baseline	2008-12-24	CEV-ICCA-TLM-docked-MOD	4200	16003	90	18665	0
3	NEAR RENDEZVOUS CEV-A-ISS – Baseline	2008-12-24	CEV-ICCA-TLM-docked-MOD	4200	16003	90	18665	0
4	Launch Ascent Post-ALAS CEV-MS – Baseline	2008-12-09	CEV-MS-TLM-ascent-MOD	4200	112986	350	1666	0
4	Launch Ascent Post-ALAS CEV-MS – Baseline	2008-12-09	CEV-MS-voice-ascent-ag1	4200	8001	358	1666	0
5	Launch Ascent Pre-ALAS CEV-MS – Baseline	2008-12-09	CEV-MS-TLM-ascent-MOD	4200	112986	350	1666	0
5	Launch Ascent Pre-ALAS CEV-MS – Baseline	2008-12-09	CEV-MS-voice-ascent-ag1	4200	8001	358	1666	0
6	Launch Countdown CLV-MS – Baseline	2008-12-09	CLV-MS-TLM-countdown-MOD	6000	112979	350	1666	0
6	Launch Countdown CLV-MS – Baseline	2008-12-09	CEV-MS-voice-countdown-ag1	6000	8001	358	1666	0
6	Launch Countdown CLV-MS – Baseline	2008-12-09	CEV-MS-voice-countdown-ag2	6000	8001	370	1666	0

Figure 5 – Constellation Communication Simulation Summary

B. Integrated DSIL Operation – Distributed Simulation

The goal of Distributed Systems Integration Laboratory (DSIL) is to support integrated testing as early as possible, to test performance and do early risk mitigation. Issues identified in this manner can be corrected early in the lifecycle, avoiding costly late-development changes and also saving time and money.

The driving requirement for DSIL is data exchange. This integration architecture involves several data interface layers that must be developed to support DSIL objectives. The architecture must support:

- Communication between physically joined elements, such as the Crew Exploration Vehicle (CEV) and the Crew Launch Vehicle (CLV). Elements exchange information along flight data buses at relatively high frequency.
- Communication between distributed, simultaneously active flight elements, including flight vehicles and operations facilities. Data is telemetered among the elements, for insight and commanding.
- Simulation state and dynamics, to ensure system performance and integration are analyzed in a consistent and realistic flight environment.
- Facility command and control data, to coordinate and monitor the execution of the distributed hardware and software involved in the tests
- Data management, to control the authoritative data sources to initialize elements and preserve test information for later analysis.

The DSIL architecture offers an accurate representation of mission operations concepts of the Constellation Program. The plan for the first mission phase is to orbit the CEV around the Earth, dock the vehicle with the International Space Station (ISS), and return the CEV to Earth. The elements that comprise the first and later missions are the CEV, CLV, Mission Control Center (MCC), Launch Control Center (LCC), and Communications and Tracking Network (CTN). Space Communications and Navigation (SCaN), which administers the Space Network (SN), Deep Space Network (DSN), and Near-Earth Network (NEN), performs communications and tracking together with the Air Force (AF) Launch Heads.

DSIL consisted of multiple test labs that were geographically distributed throughout the United States. These test labs were connected through NISN. The NISN backbone and the externally accessible IP addresses through the facility network form the NASA Distributed Simulation network (DSNet). The following figure shows the eight NASA centers connected through the DSNet: Ames Research Center (ARC), Goddard Space Flight Center (GSFC), Glenn Research Center (GRC), Jet Propulsion Laboratory (JPL), Johnson Space Center (JSC), Kennedy Space Center (KSC), Langley Research Center (LaRC) and Marshall Space Flight Center (MSFC).

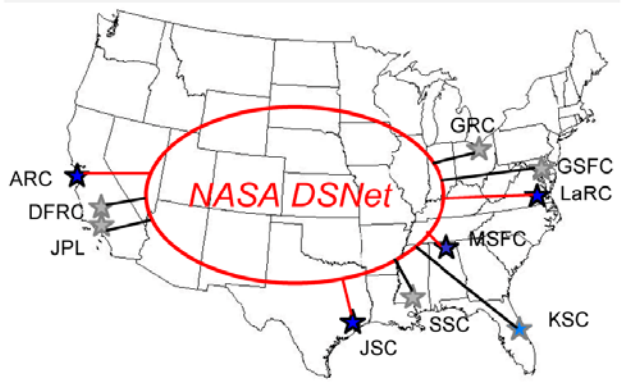

Figure 6 – Distributed Simulation Network (DSNet)

HLA provides the infrastructure for synchronization and control of the various simulations. An association of possibly distributed processes cooperating using HLA is called a federation where each process participating in the federation is a federate. A federation is usually created by the first process (federate) accomplishing the dynamic registration process (i.e. 'joining') the federation and destroyed by the last process leaving the federation. Federates exchange information through a publish/subscribe mechanism to update object attributes. Interactions among federates are realized through sending and receiving interaction events. A federation has a specific federation object model (FOM) that defines the structure of the types of objects and interactions that may be exchanged by participating federates. A runtime infrastructure coordinates the processes so that federates in a federation can be time-synchronized.

The SCaN federate uses an IP Network Emulator (IPNE) to handle incoming and outgoing packets. IPNE implements a packet sniffer/injector in conjunction with the external interface API. It sniffs packets from the physical network, sends packets through the QualNet simulation, and injects them back into the physical network. With the IPNE interface, the SCaN federate sniffs the IP packets on the port interface and filters the packets that are coming from either CEV or CLV and destined for the MCC software. These packets are injected into the SCaN NI&E Simulator. Inside the simulator, there are two virtual nodes representing CEV and CLV that receive the injected packets. Communications and protocol effects are simulated as the packets "pass through" the TDRSS bent pipe to WSC. Our link budget library is called to compute the bit error rate / frame error rate (BER, FER) of the bent-pipe link. Depending on BER and FER, data may be dropped. The data that is not dropped are then relayed to a virtual node in the SCaN NI&E Simulator representing MCC, where IPNE takes these packets and injects them back into the physical network destined for MCC. The packets go out of a port using UDP onto DSNet to the MCC software (at JSC) that reads from UDP socket.

Before each DSIL test/demo, there had to be an agreed upon configuration defining the specific IP addresses and port numbers to use at each of the participating NASA centers. Each partner must be running the same version of the HLA real-time infrastructure software and the same FOM.

There had been four DSIL Demos.

- *Dec 2007 Demo*: the scenario was CEV to ISS mission, Ascent phase (8 minutes). In this demo, we tested data exchange among 5 major system facilities (CEV, CLV, Launch Control Center (LCC), Mission Control Center (MCC) and SCaN). HLA was used to synchronize time steps and exchange simulation truth data (spacecraft positions, speed, attitude, etc.) Another program, Mission Automation Environment for Simulation, Test, and Real-time Operations (MAESTRO, developed at MSFC) was used to coordinate C3I telemetry data flow (e.g. CLV and CEV telemetry) among the facilities through UDP sockets. Telemetry data flows were relayed through SCaN to MCC
- *Feb 2008 Demo*: this demo is similar to Demo 1, with the following additions. For situational awareness at all centers, Distributed Object Network (DON) was used to show the graphics of CEV from launch pad to ascent, and CLV separation. This demo also includes the SCaN emulator in passive listening mode and other DSIL IU tests.
- *June 2008 (joint IMSim/DSIL) Demo*: the general purpose of this joint DSIL/IMSim test was to execute a distributed simulation of CEV/CLV from pre-launch through ISS docking.

- *Feb 2009 Demo*: the plan for this demo was to have the SCaN Emulator emulate the bent-pipe between CEV and WSC, while the SCaN NI&E Simulator simulates the link between CLV and Air Force TEL4 ground station. In this demo, CLV traffic was simulated by GSFC. The SCaN NI&E Simulator received simulated CLV UDP/IP traffic data streams from an external CLV source (GSFC) and relayed the data to TEL4.

IV. Conclusion and Future Work

In this paper, we described the need for developing a network simulation tool for space networking. In particular, leveraging on a commercially available network simulation tool, QualNet, we added custom models that are necessary to simulate a mission where CEV flies to the ISS and returns to Earth. Although the Space Shuttle had already flown to the ISS and back, the new Constellation missions will use IP-based protocols in space. Thus, there is the need to evaluate the network performance using IP-based protocols over space data links. The custom models needed for this mission scenario are: CCSDS ENCAPS and CCSDS AOS (M_PDU). To do an end-to-end simulation, we need to be able to generate data traffic representing mission data. We also need to model the unique physical characteristics of the communications system. The SCaN NI&E Simulator is the result of these customizations. The tool has been used to simulate nominal mission scenarios; it has also been used in distributed simulation in a Distributed Simulation Integration Laboratory. As NASA's space networking architecture continues to evolve, new capabilities will be identified that need to be modeled in simulation. A few areas being investigated are: Quality of Service (QoS), security (e.g. at network layer, link layer), and the use of CCSDS Space Link Extension services. New information on the spacecraft's antenna and radio may need to be accurately modeled in the simulator. We are also adding more detailed Space Network Operations modeling into the tool. We also envision new suite of protocols need to be evaluated for Lunar missions and Mars missions.

Acknowledgments

Part of the work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] Linsky, T., Bhasin, K., White, A. and Palangala, S., "Simulation of Lunar Surface Communications Network Exploration Scenarios", *AIAA Modeling and Simulation Technologies Conference*, Aug. 2005.

[2] Jennings, E., and Heckman, D., "Performance Characterization of Space Communications and Navigation (SCaN) Network by Simulation", *IEEE Aerospace Conference*, March, 2008.

[3] CxP 70078 Draft A, "Constellation Program Computing Systems Architecture Description Document (CSADD)", Rev. A, 21 March 2008, (280 pages).

[4] CxP 70022-03 "Command, control, communication, and information (C3I) interoperability standards book, volume 3: master link book", C3I20Vol3 , revision A.

[5] Consultative Committee for Space Data Systems (CCSDS), "AOS Space Data Link Protocol", Recommended Standard, CCSDS 732.0-B-2, July 2006.

[6] Soloff, J., "Cx-SCaN Architecture Update CR Process" presentation, February 28, 2008.

[7] Cheng, M., "Frequently Asked Questions on Low-Density Parity Check (LDPC) Codes", presentation to SCIP SE&I, May 9, 2007.

[8] Consultative Committee for Space Data Systems (CCSDS), "Encapsulation Service", Recommended Standard, CCSDS 133.1-B-1, June 2006 (with Technical Corrigendum dated October 2006).

[9] Stoenescu, T., and Clare, L., "Traffic Modeling for NASA's Space Communications and Navigation (SCaN) Network", *IEEE Aerospace Conference*, March, 2008.

[10] "Best Practices: NetIQ Analysis Center for VoIP – A White Paper for VoIP Quality", http://download.netiq.com/CMS/WHITEPAPER/ReportingBestPractices_VoIP_VoIPQuality (Oct 2005).pdf