A CASE STUDY OF HIGH-VOLUME AUTOMATED TESTING WITHIN THE NASA SPACE NETWORK

Dana Irvin, Kirill Lokshin, Amit Puri
Ingenicomm, Inc.
14120 Parke Long Court #210, Chantilly, VA 20151
dirvin@ingenicomm.com

ABSTRACT

The NASA Space Network (SN), which consists of the geosynchronous Tracking and Data Relay Satellite (TDRS) constellation and its associated ground elements, is a critical national space asset that provides near-continuous, high-bandwidth telemetry, command, and communications services for numerous spacecraft and launch vehicles.

SN sustainment activities frequently involve testing of the numerous interfaces within the SN ground segment. To reduce the cost and complexity of such testing, NASA commissioned the development of the External Bearer Interface Test Set (XBIT), which enables ground interface verification using a high-volume test automation framework.

This paper considers the use of the XBIT as a case study of automated ground segment verification and validation. The paper discusses the trade-offs between automated, semi-automated, and interactive ground interface testing and presents comparative test execution metrics to quantify the relative efficiency of these approaches.

INTRODUCTION

The NASA Space Network (SN) was established in the early 1980s to replace NASA’s worldwide network of ground tracking stations, with increased footprint and availability for space missions. It consists primarily of the geosynchronous Tracking and Data Relay Satellite (TDRS) constellation and its associated ground elements, and provides near-continuous, high-bandwidth telemetry, command, and communications services for numerous missions, spacecraft and launch vehicles. It operates as a bent-pipe relay system between various customer platforms and their ground facilities (as shown in Figure 1).
The SN involves multiple ground sites, including operation ground terminals, as well as test and support facilities. The primary ground terminals operate at NASA’s White Sands and Guam sites, and serve as the bridge point between the TDRS constellation and NASA’s Integrated Services Network (NISN). The NISN facilitates the terrestrial communications links between the SN ground assets and the customer platform’s mission operations centers (MOCs), as well as other customer ground facilities.

Data exchanges between the NISN and customer facilities utilize numerous protocols and data formats, depending on the type of data been transmitted. Telemetry and command data, data to the customer MOC and from the customer MOC, respectively, utilize either Space Link Extension (SLE) services defined by the Consultative Committee for Space Data Systems (CCSDS) or one of several older NASA network services, such as the Internet Protocol Data Unit (IPDU) or Low Earth Orbit-Terminal (LEO-T) protocols.

The SN ground segment is undergoing a significant, enterprise-wide upgrade via NASA’s Space Network Ground Segment Sustainment (SGSS) program. This program will replace various subsystems responsible for customer MOC interfaces with a single user services element, which will implement both legacy and modern MOC interface protocols. In order to perform validation and verification (V&V) testing of the customer interfaces during the SN/SGSS user services element development, as well as during sustainment as it transitions in to operations, NASA commissioned the development of the External Bearer Interface Test Set (XBIT), a consolidated data simulation and interface emulation system capable of testing each of the affected interfaces.

**XBIT SYSTEM DESIGN**

The XBIT system was designed to serve as an end-to-end simulation and emulation tool for digital baseband and network interface V&V testing through SN/SGSS development and sustainment. In order to perform that function, the XBIT incorporates three principal data generation and processing subsystems:
1. Network gateways, equipped with Gigabit Ethernet and 10-Gigabit Ethernet interfaces. Each XBIT network gateway is configured to support up to 32 full-duplex network links, all of which can be independently configured and operated.

2. High-rate front-end processors (FEPs), equipped with ECL serial clock-and-data interfaces. Each XBIT high-rate FEP is configured to provide two serial transmit and two serial receive interfaces, all of which can be independently configured and operated.

3. Low-rate FEPs, equipped with RS-422 serial clock-and-data interfaces. Each XBIT low-rate FEP is configured to provide four serial transmit and four serial receive interfaces, all of which can be independently configured and operated.

The three types of XBIT subsystems utilize a common digital data processing software framework, which implements the various spacecraft and ground data simulation and processing capabilities and protocols. These protocols include CCSDS Telecommand (TC), Telemetry (TM), and Advanced Orbiting Systems (AOS) services. The MOC interface emulation capabilities include the modern CCSDS SLE services, including Return All Frames (RAF), Return Channel Frames (RCF), Return Operational Control Field (ROCF), Forward CLTU and Enhanced Forward CLTU (EFCLTU) services, as well as the legacy NASA network communications formats, including ACE, LEO-T, SFDU, NASCOM and IPDU.

In addition to the data processing software, the XBIT system includes a management and automation framework. This framework provides system monitoring and control functions; however, it primarily serves as the test automation framework, providing support for test scenario definition, execution, and reporting. This automation framework will be discussed in more detail in later sections.

In order to adequately perform its V&V function, the XBIT system must be able to be used to emulate both the MOC interfaces as well as the spacecraft interface. Representative data flows for these scenarios are shown in Figure 2.
In a space-to-ground (downlink) simulation scenario (shown in the left portion of Figure 2), the XBIT generates a simulated spacecraft data stream at baseband, which may be further encoded with forward error correction (FEC) symbols. The data is then transmitted into the SN/SGSS digital signal processing element (labeled “A”). Alternatively, the un-encoded signal may be sent directly to the user services element (labeled “B”), and should be identical to the expected output of the SN/SGSS digital signal processing element in the “A” scenario. For either scenario, the resulting data is sent back to XBIT for verification using the MOC emulation protocol of choice (SLE or legacy, labelled as “C”).

In a ground-to-space (uplink) simulation (shown in the right portion of Figure 2), the XBIT sends a user data signal via the MOC emulation protocol of choice (SLE or legacy) to the user services element (labeled as “C”). The user services element either sends the data directly to the XBIT (labeled as “B”) or sends it for encoding by the SN/SGSS digital signal processing element prior to transmission back to the XBIT for verification (labeled as “A”).

In either simulation scenario, the XBIT system operates as both the simulated data source, as transmitted through the SN/SGSS ground element, as well as the data’s ultimate destination. This source/destination paradigm allows for the verification of protocol conformances on the tested interfaces, as well as verification of data integrity as it flows through the system.

AUTOMATION MODEL

As previously stated, the XBIT system is designed to perform comprehensive V&V testing of multiple interfaces, protocols, data sizes, and other encoding formats. The XBIT system is capable of executing tens of thousands of various testing scenarios, many simultaneously or independently. This testing can encompass three different modes of operation: interactive, semi-
automated, or fully automated. These three modes operate on one or more layers of the XBIT Automation, as shown in Figure 3.

![Figure 3: XBIT Automation Model](image)

**Test Execution Layer and Interactive Testing**

The test execution layer of the XBIT automation module is provided by the commercial off-the-shelf CGS digital data processing software package. This package is responsible for performing the underlying operations and data exchanges which comprise each test scenario.

The test execution layer provides the ability to perform interactive testing. As there are tens of thousands of possible test scenarios needed to perform comprehensive system V&V testing, interactive testing is impractical as the major methodology for SN/SGSS testing. For example, if 10,000 test scenarios were to be manually executed in this mode, with each scenario taking 10 minutes, one test engineer would need the better part of a year to execute each test – once. However, the interactive testing mode does provide the ability to perform spot checks, as well as allowing the test engineers to further refine automation test scenarios. Test scenario refinement is the most common driver for using this layer and mode of operation.

**Test Automation Layer and Semi-Automated Testing**

The test automation layer of XBIT is implemented by a software application called “icAutomata”. This software package is responsible for translating the parameters and steps defined in each test scenario into subsystem-specific configuration values and operations.

The icAutomata software communicates with each XBIT subsystem using web services (REST and Websockets) APIs provided by the CGS digital data processing and generation software. Through these APIs, icAutomata is able to address every data generation and processing software instance, configure individual software parameters, start and stop data flows, and monitor and evaluate the status of the software and its component elements.
Each test scenario defines a set of verification points, which are collected for each execution of a scenario when the test scenario is performed. The verification points may be simple numerical statistics, such as the number of bits transmitted or the number of pattern errors detected, or more complex Boolean criteria, such as the results of a bit-level comparison between input and output files. Once a test scenario has been configured and data flow begins, the icAutomata software monitors the various status indicators and statistics generated by the data processing software, and collects the verification points as defined in the test scenario. The collected verification points are used to generate reports for the test program, which not inform the test engineer that a test passed or failed, but can also provide a detailed test record by including the verification statistics as part of the report.

In the XBIT system, the icAutomata layer is considered to provide the “semi-automated” testing capability. The reason for this distinction is that icAutomata does not generate and execute a complete sequence of test scenarios; rather, it is used to execute a single scenario at a time. Despite this, semi-automated testing offers obvious efficiencies when compared to purely interactive testing; while interactive testing may take 10 minutes to execute a given test scenario, with some amount of time needed to switch configurations between tests, the average execution and switching times are significantly reduced when each scenario is pre-configured and automatically executed through icAutomata.

**Test Orchestration and Fully Automated Testing**

The test orchestration layer of the XBIT automation is implemented by a software application call “SKYNET”. This package is responsible for selecting, configuring and invoking the execution of specific test scenarios based on a user-defined test program.

The XBIT test orchestration layer relies on a set of pre-defined base test scenarios defined in the semi-automated testing layer. These baseline test scenarios are stored in a configuration-controlled repository. During the execution of an automated test program, the test orchestration software retrieves one or more base scenarios from the repository and dynamically adapts them to generate a series of derived scenarios, which are individually invoked. This process is shown in Figure 4.
The adaptation of the base test scenario is a programmatic procedure that induces controlled variation into for a series of individual parameters in said base scenario. For example, consider a simple network bit error rate test (BERT) scenario, where one subsystem generates fixed-size data packages containing a pseudo-random binary sequence (PRBS) and transmits those packages to a second subsystem via TCP/IP. A second subsystem receives this package and verifies the data. This scenario is shown in Figure 5.

While this example scenario is very simple, and limits potential variation, different test configurations could be achieved by adjusting the data generation rate or frame size. A potential variations set is shown in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit Rate</td>
<td>1 kbps</td>
<td>900 kbps</td>
<td>+1 kbps</td>
</tr>
<tr>
<td>Frame Size</td>
<td>100 bytes</td>
<td>4000 bytes</td>
<td>+1 byte</td>
</tr>
</tbody>
</table>

Table 1: Potential Parameter Ranges/Steps for BERT Scenario
Given the above parameter ranges, if the test orchestration software were to exercise each of the possible combinations, it would generate and execute more than 3.5 million derived test scenarios. If this level of exhaustiveness in testing is required, it can be performed using XBIT’s dynamic test adaptation approach. The dynamic nature of the derivation means the manual effort to set up such a test program is fixed, and involves only the configuration of the base scenario and the selection of the parameter variations; increasing the parameter range or decreasing the step size does not create additional work for the test operators.

Additionally, whenever a base test scenario defines verification points, each execution of a derived scenario also collects these verification points as part of the executed test report. These reports can then be used to provide even greater visibility into the testing process, providing exceptionally detailed information regarding the completeness of testing.

**TEST APPROACH AND RESULTS**

The SN/SGSS approach to automated testing using XBIT is structured around a nightly testing cycle. This is done to accommodate typical schedules of staff, as well as to allow semi-automated or interactive testing, scenario refinement, and other test support activities during normal business hours. Nightly testing activities are planned in a sequence that facilitates this “lights-out” approach.

All four XBIT subsystems (two network gateways, one high-rate FEP, and one low-rate FEP) are utilized during each nightly testing cycle, with individual scenarios using one or more subsystems. Additionally, multiple instances of the data processing software on each subsystem can be used, depending on the specific interfaces to be exercised. The highest utilization achieved by single test scenario includes all four subsystems and a total of discrete 18 software instances.

The specific selection of test scenarios to be executed during a given nightly run is based on SN/SGSS program needs. Priority is often given to tests that involve new or modified ground element interfaces and/or functions, as well as interfaces/functions which are more frequently needed by customer missions.

The SN/SGSS test scenario library utilizes 26 different interface and service configurations, ranging from a single low-rate serial (RS-422) interface up to a combination of four low-rate serial (RS-422) interfaces, two high-rate serial (ECL) interfaces, and eight packetized network services. Additionally, the scenarios exercise service data rates ranging from 100 bps to 500 Mbps, and employ a number of different payload data types, including CCSDS TM, TC, and AOS service data units, non-CCSDS structured data packages, and unformatted contiguous bitstreams.

During a recent four-month test window, the XBIT system executed more than 36,000 individual automated test scenarios, as shown in Table 2.
<table>
<thead>
<tr>
<th>Service/Interface Configuration</th>
<th>Scenarios Executed</th>
<th>Data Volume Processed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-rate serial</td>
<td>13,000</td>
<td>450 GB</td>
</tr>
<tr>
<td>High-rate serial</td>
<td>5,600</td>
<td>8 TB</td>
</tr>
<tr>
<td>Network only</td>
<td>18,000</td>
<td>16 TB</td>
</tr>
</tbody>
</table>

Table 2: Four-month Test Window Execution and Data Volume Statistics

The XBIT system continues to support SN/SGSS interface testing and is allocated for continuous use through the end of 2017.

**CONCLUSIONS**

Based on the results achieved by SN/SGSS, the multi-layer test automation model adopted by XBIT has proved to be an effective method for exhaustive interface testing and verification. By allowing the dynamic adaptation of test scenarios from a controlled baseline, the XBIT test orchestration approach allows NASA to rapidly define and execute a large number of individual test scenarios, which can exercise different configurations and parameters on any particular interface. This, in turn, allows NASA to easily conduct consistent and thorough testing of the SN ground elements as they are modernized, providing assurance of interface interoperability following changes while simultaneously reducing the total cost of interface testing and verification throughout the lifetime of the program.