Emulated Spacecraft Communication Testbed for Evaluating Cognitive Networking Technology

Joseph Downey, Adam Gannon, Aaron Smith, Mick Koch and Rachel Dudukovich
Cognitive Signal Processing Branch
NASA Glenn Research Center
Cleveland, OH
Email:{joseph.a.downey, adam.gannon, aaron.smith, mick.v.koch, rachel.m.dudukovich}@nasa.gov

***Abstract*—The ability to emulate the full space protocol stack is an essential aspect required to evaluate and mature cognitive communication capabilities. The interaction between the physical layer and network layers is key to developing network optimizations for a dynamic and complex environment. We present a laboratory testbed for the evaluation of cognitive radio and networking techniques applied to space communications. The testbed is a high fidelity, flight-like hardware testbed consisting of software-defined radios, channel emulators, modems, and orbital analysis and scheduling software. The testbed uses RF links with signal quality, propagation delay, and Doppler effects driven by orbital mechanics simulations of emulated spacecraft. Our framework enables control of link bidirectionality, data rates, and interference sources. In addition to hardware radio nodes, the testbed can incorporate virtualized emulated nodes for larger and more challenging network scenarios. Our approach to a cognitive communication system uses delay tolerant networking (DTN) to mitigate the challenges of the space environment. While many DTN networks use only preplanned schedules, our system uses User-Initiated Service (UIS) to dynamically schedule service providers. Software-defined radio allows the system to adapt to a variety of service providers. Integration of DTN, UIS, and software-defined radio technologies provides a framework for the implementation of a cognitive communication system. This paper describes the testbed capabilities, network emulation approach, component integration, and initial end-to-end testing results.**

***Index Terms*—cognitive radio, network emulation, RF testbed, user-initiated service, delay tolerant networking**

I. Introduction

NASA's space communication networks have been successfully providing service to user missions for many decades, enabling scientific discoveries and expanding human exploration. In the coming decades, NASA and partner space agencies will continue to explore the Earth, Moon, Mars and beyond. The anticipated communications infrastructure required to support future human and robotic missions will require greater degrees of autonomy, flexibility, resiliency, interoperability, and performance. Automation will be required to reduce human effort and operating costs, especially considering the distance from Earth and the associated latency. Communication links will be automated, dynamically changing link parameters in response to channel conditions. Flexible scheduling approaches will allow missions to request services on-demand, enabling rapid response to scientific opportunities. As NASA transitions from government-owned to commercially-provided communication services, there is a need for seamless roaming and interoperability between multiple disparate providers. Finally, network optimizations will be needed to efficiently handle the various levels of Quality of Service (QoS) associated with high-resolution images/video, health/status, and instrument data supporting human exploration on the Moon and Mars.

NASA's Cognitive Communications Project has been researching and developing prototype cognitive communication systems to address the challenges of complex and dynamic space environments. Cognitive communication technologies mitigate the complexity by increasing the autonomy of all aspects of a communications system, including the link/networking layers, service scheduling, and network management. Our approach uses delay tolerant networking (DTN) to provide resilient and reliable communications in the midst of intermittent link connectivity, long delays, and data-rate mismatches. While many DTN networks use only pre-planned schedules, our system uses User-Initiated Service (UIS) to perform dynamic scheduling with service providers. Furthermore, the service execution is automated and will autonomously configure the spacecraft's communication terminal, networking software, and corresponding mission operations center (MOC) software without human interaction.

This paper presents a laboratory testbed for the evaluation of cognitive radio and networking techniques applied to realistic mission scenarios. The interaction between the physical layer and network layers is key to developing network optimizations for a dynamic and complex environment. By using RF links with physics-accurate signal quality, delay, and Doppler effects, the testbed provides a high-fidelity emulation for evaluating and maturing automated/cognitive capabilities. Data collected on this testbed can be used to baseline current systems and demonstrate improvements of automation and cognitive networking technology in a relevant environment.

This paper is organized as follows: Section II provides an overview of the emulation testbed and detailed description of

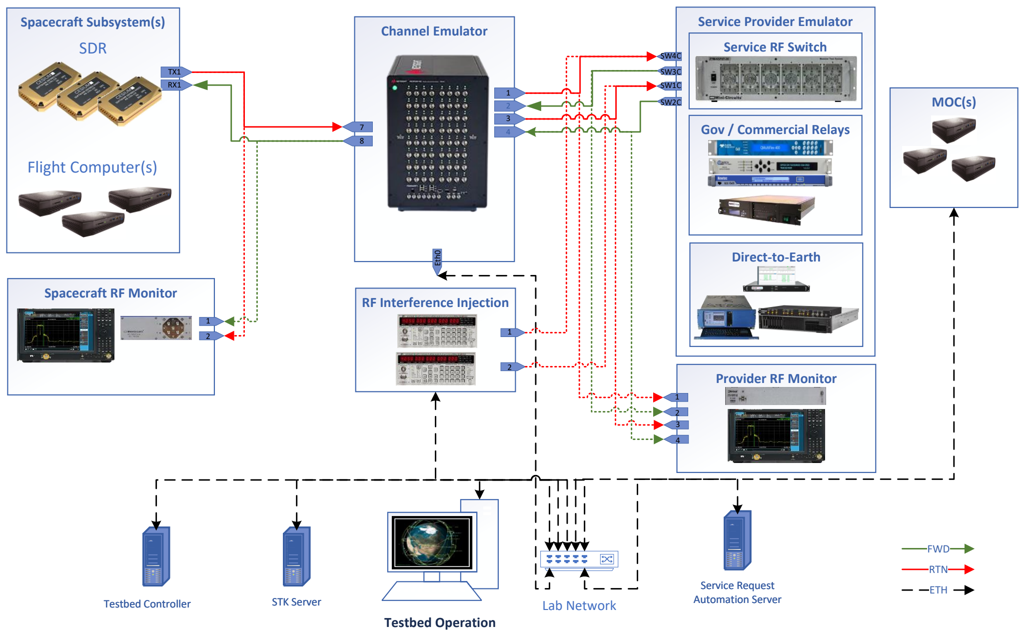

Fig. 1. Overview of major testbed components. Red and green lines represent Forward (FWD) and Return (RTN) link signals, which can also be uplink and downlink respectively for direct-to-Earth services. Red and green dotted lines represent coupled ports for monitoring or interference injection.

the components. The concept of operations for the testbed is provided in Section III, describing the intended use of the testbed, data flows, and system metrics. Finally, in Section IV, results from end-to-end testing is provided for an example scenario. Concluding remarks and next steps are discussed in Section V.

II. Testbed Overview

The testbed provides a high-fidelity emulation environment for evaluating the end-to-end performance of a communication system. It consists of software-defined radios (SDRs), flight computers, channel emulators, modems, orbital analysis, mission operations, and network automation servers. Where appropriate, flight-like hardware is utilized to demonstrate feasibility on SWaP-constrained platforms. The current implementation of the testbed is shown in Fig. 1 and the following subsections provide an overview of the primary components.

A. User Spacecraft Emulation

Currently, the user spacecraft emulation focuses of the communication subsystem and the flight computer data handling aspects of the mission. The communication subsystem consists of a SDR, as shown in Fig. 2. Preliminary testing has only required one or two emulated user spacecraft thus far, however, additional nodes are available for multi-hop routing and multi-spacecraft constellations. The following sections provide further details on the hardware and software of the emulated user spacecraft.

1) Software-Defined Radio: The user spacecraft is assumed to have one or more re-configurable wideband software-defined radios (SDRs) which can be loaded with appropriate waveforms compatible with each service provider [1]. To represent this wideband terminal, a CesiumAstro SDR-1001 engineering model was selected. The radio is a high-performance, low SWaP compact design designed for Low Earth Orbit (LEO) environments and is tunable from 300 MHz to 6 GHz with 100-200 MHz of bandwidth [2]. The SDR includes a set of 4 independent transmit/receive pairs for flexible multi-antenna operations. The companion CesiumAstro single board computer SBC-1461 intefaces with the SDR and the flight computer and is also available for additional data processing.

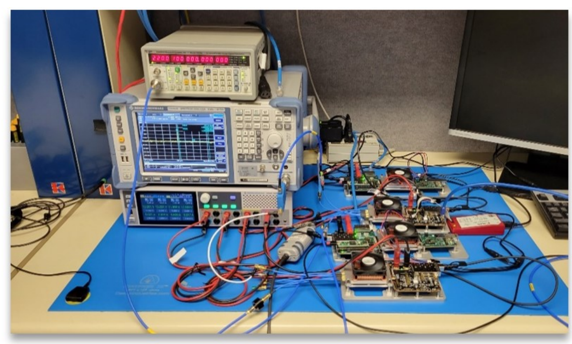

Fig. 2. User spacecraft emulation subsystem, consisting of SBC, SDRs, and associated power supplies and instrumentation.

A set of modem applications (“waveforms”) was developed for compatibility with a variety of commercial and government service providers. Some waveforms were purchased as intellectual property, while others are available from NASA’s software catalog. Table I provides a summary of the waveform applications, along with the intended service provider type.

Waveform	Modulation	Max. Rate	Function
Spread Spectrum	SS-BPSK	193kbps	Gov SR
QPSK	QPSK	31Mbps	DTE-S
DVB-S2	PSK, APSK	31Mbaud	Com SR
HRBE	PSK, APSK	83Mbaud	DTE-Ka

TABLE I
WAVEFORMS FOR SPACE RELAY (SR) AND DIRECT-TO-EARTH (DTE)

2) Flight Computer: The flight computer emulates the software components of a generic science mission. Custom scripts are used to automate a variety of file transfer patterns, representative of data files generated by a science instrument, spacecraft health and status log files, or other user information. The system models several classes of data by creating variable data rates and patterns (bursty vs continuous), differing file sizes, and data priorities.

Several aspects of the application layer components have been heavily influenced by delay tolerant networking (DTN) concepts [3]. This approach was selected for several reasons. DTN has been featured on several missions that are comparable to the testbed's LEO scenario, such as NASA's Plankton, Aerosol, Cloud, ocean Ecosystem (PACE) [4] mission, the International Space Station (ISS) [5], and ESA's OPS-SAT mission [6]. The approach provides a set of protocols that are being standardized by organizations such as the Consultative Committee for Space Data Systems, which will allow for interoperability among many entities (multiple government agencies, industry, and academia). Many concepts required for space communication such as store-and-forward, contact plans, data priority, data encryption and reliable transport have already been well established within the DTN framework, and easily adopted without the need to develop custom solutions.

The testbed utilizes High-rate Delay Tolerant Networking (HDTN) as its DTN implementation [7], [8]. HDTN provides the file transfer applications, storage management, statistics logging, web interface, Bundle Protocol [9] and Licklider Transmission Protocol [10] that are used on the testbed. The data priority scheme is based on the class of service described in RFC 5050 (00 = bulk, 01 = normal, 10 = expedited), where higher values indicate a higher priority. HDTN's storage module manages both priority and data expiration time, ensuring high priority data will not expire without being sent, in preference over low priority data. The user is responsible for establishing the priority and lifetime of the data, which is configured via the file transfer application.

Licklider Transmission Protocol (LTP) was selected as the transport protocol beneath the bundle layer. The ISS DTN network, [5], the Laser Communications Relay Demonstration [11], and the LunaNet specification [12] have recommended LTP for space-to-ground links. LTP provides reliability through block acknowledgment and retransmission. LTP is suitable for links with long delays, minimizes overhead by using selective negative acknowledgments (NAK) and aggregates client service data units into larger blocks for acknowledgement. In addition, the HDTN implementation of

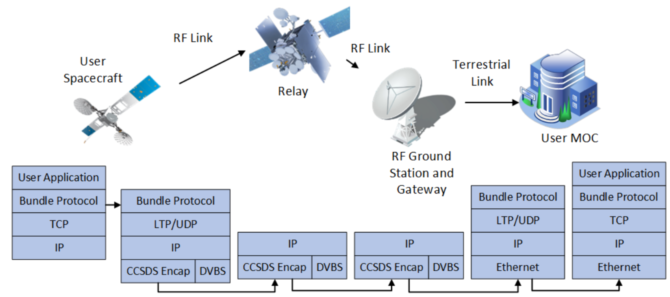

Fig. 3. Protocol stack for emulated near-Earth scenario

LTP includes an "LTP ping" capability in which the sender transmits an LTP cancel segment of a known non-existent session number to the receiver during times of inactivity. This causes the receiver to respond with a cancel ACK, which allows HDTN to sense if the link is still active. LTP is built on top of User Datagram Protocol. The testbed uses IP packets to maintain compatibility with a variety of applications and commercial providers. Fig. 3 shows the Near Earth scenario protocol stack.

B. Channel Emulation and Orbital Mechanics

The central component of the testbed is the channel emulation subsystem and the associated orbital mechanics analysis and modeling. The function of the channel emulator subsystem is to provide realistic communication impairments from the environment such as atmospheric effects, ground or satellite-based RF interference, AWGN, Doppler, multi-path, and delay. The orbital mechanics model keeps track of the location of every asset in the simulation, and provides the resulting link parameters to the hardware channel emulator for every active communication link. If needed, the simulation will also be used to determine atmospheric attenuation in conjunction with an attenuation model (e.g. ITU propagation model).

The channel emulation hardware consists of a Keysight Propsim F64 Radio Channel Emulator, outfitted with 24 pairs of transmit/receive ports. Although the actual service may be operating in L-,S-,X-,Ku-, or Ka-band, the testbed operates at a common intermediate frequency (IF). The channel emulator applies the appropriate impairments as if the signal were at the intended operational frequency. The emulator also can perform frequency translation, enabling the spacecraft SDR to be compatible with a broad range of modems.

The testbed primarily uses AGI's System Tool Kit (STK) for modelling the environment and orbital dynamics. Other orbital dynamics software applications are also suitable, however the programming interface of STK enables automated operations of the testbed. The orbital mechanics simulation will be used to determine if links between multiple nodes are possible. When a link is possible, it will be used to determine the potential impairments of the communication link. Those impairments will then be programmed into the channel emulator and external interference sources.

RF interference is currently injected after the channel emulator using a Vector Signal Generator which can generate arbitrary interference waveforms. The interference can be remotely controlled and triggered via a control interface. Alternatively, an interfering satellite could be added as an additional input to the channel emulator and modeled in the orbital dynamics software. Currently, only the first method has been tested operationally.

C. Service Provider Emulation

Each service provider has a set of ground stations or relay satellites that are used to communicate with the user mission. The overall attributes (location, effective isotropic radiated power (EIRP), G/T) of those assets are modeled by the orbital analysis tool. In the current implementation of the testbed, the relay satellites are assumed to be in a bent-pipe configuration with the effective overall channel modeled as the cascade of both links. To emulate the unique features of each service provider, representative modems were procured for each service, as highlighted in Fig. 1.

D. Service Request Automation Server

The Service Request Automation Server (also referred to as the UIS Server) enables automatic scheduling and provisioning of satellite service [13]. The process begins with the UIS client application on the user spacecraft generating a request message conveying the destination, data volume, and latency constraints. This message can be sent over low-rate control channel, or an in-band message part of an already scheduled data transfer. The UIS server processes the request and determines an appropriate service provider. While the UIS server can actually schedule communication services from several providers, it also has emulation modes for operating the testbed which emulate the response of the service provider. A short response message is sent back to the spacecraft to acknowledge the service request and confirm the approved events.

E. Testbed Controller

Propsim channel emulations are built from a file-based graph description, typically using the onboard graphical editing interface, so some additional software was required to enable automated channel emulations for the testbed. The Testbed Controller implements this automated control of the Propsim and other components (e.g., RF switches) for testbed operations. The remainder of this section provides an overview of how these channel emulation events are processed.

To start, the controller listens for scheduled access messages sent by the event scheduling service. These messages describe the event's start and stop times, as well as the assets scheduled to communicate and the waveform configuration they will use. The controller uses this information to request the event's channel models from the STK scenario service and to schedule the event for later processing. These upcoming events are tracked by the controller in time-based queues, and events that overlap in time are merged into event groups. Prior to each event group's scheduled start time, the controller generates and uploads the emulation files required by the Propsim and then directs the Propsim to compile these files into its proprietary emulation format. The Propsim is currently used in aerospace and satellite option (ASO) channel mode, so the generated files include the necessary ASO channel model files as well as a top-level emulation file (SMU). The ASO files provide a sequence of delay, Doppler, and attenuation values to apply to each channel over time, and the SMU file describes the graph of input and output ports and the channels connecting them. Finally, when the scheduled start time is reached, the controller loads and starts the emulation on the Propsim and updates the testbed radio frequency (RF) switches to route the relevant assets through the Propsim's ports. The testbed assets included in the access event are then able to communicate during the access window using a channel with realistic delay, Doppler, and attenuation effects.

III. Concept of Operations

The testbed allows radio pairs to communicate with each other over an emulated wireless channel by managing a simulated physical environment. A directory of CSV files is used to formally define a scenario, providing the testbed components with information such as the spacecraft orbits, ground station locations, high-level RF definitions (i.e. G/T and EIRP), radio waveforms, and user data generation profiles. During operations, users will generate data, which is then passed to their SDRs and transmitted to representative service provider modems through a channel emulator at carrier. The channel emulator applies attenuation, delay, and Doppler offsets, according to the simulated physical environment. Similarly, service provider modems are used to transmit control messages to the user spacecraft. Several metrics (see Table 1) are collected for both real-time visualization and to create a post-analysis report on system performance.

A. Scenario

Scenario definitions are used to initialize testbed components and assist with version controlling test operations for reproducibility. In large part, scenario definitions describe the simulated physical environment, such as spacecraft orbits, ground station locations, and high-level RF parameters such as transmitter EIRP. There are multiple independent orbit propagators that operate concurrently during a test, namely those in the STK Server, the UIS Server, and on the user spacecraft. The common directory helps to ensure that these independent systems agree on the state of the simulated environment. Additional details of the scenario files and management of the simulated environment can be found in [14].

Beyond the simulated physical environment, the scenario files also define user data generation profiles. Each user can have an arbitrary number of data generation processes, which are either continuously generated with a constant rate, or randomly generated using an exponential random variable for the delay between data generation events and a separate exponential random variable for the size of the data generated

Module	Metric	Location
Ingress	Total bytes sent	Sender and receiver
Ingress	Bytes sent to egress	Sender and receiver
Ingress	Bytes sent to storage	Sender and receiver
Storage	Used space (bytes)	Sender and receiver
Storage	Free space (bytes)	Sender and receiver
Storage	Bundle bytes on disk	Sender and receiver
Storage	Bundles erased	Sender and receiver
Storage	Bundles rewritten from failed send	Sender and receiver
Storage	Bytes sent to egress cut-through	Sender and receiver
Storage	Bytes sent to egress from disk	Sender and receiver
Egress	Data rate (Mbps)	Sender and receiver
Egress	Total bytes sent successfully	Sender and receiver
Egress	Total bytes attempted	Sender and receiver
Receive file	End-to-end latency	Receiver

TABLE II
DTN METRICS COLLECTED DURING TESTBED OPERATION

during an event. These data generation processes are also associated with a priority field that is used in the DTN bundle. Additionally, the scenario files are used to define the waveforms that the available to each user radio.

B. System Metrics

Several system metrics are collected via the HDTN telemetry and logging module. The three major areas where statistics are collected are data ingress, storage, and egress. Table II shows the metrics that are logged during testbed operation. Comma-separated value (CSV) files are used to store the metrics for analysis after the emulation has completed.

IV. END-TO-END TEST

For an initial demo of testbed capabilities, we emulate the communications from a satellite carrying out science mission objectives in a medium-inclination (51°) low-Earth orbit. In this scenario the mission is serviced by two relay satellite constellations: NASA's Tracking and Data Relay Satellite System (TDRSS) and Inmarsat's Global Xpress (GX). The two constellations provide continuous high-rate (∼Mbps) coverage to the spacecraft but their services must be scheduled at least several minutes in advance. Additionally, TDRSS provides a continuous low-rate (∼kbps) link to the spacecraft. The spacecraft's instruments generate volumes of high-priority science data with little advance notice (e.g. observations of severe weather or transient astronomical phenomena) whose value to data users decreases rapidly with time. Furthermore, links are subject to intermittent disruption due to equipment failure or unintentional interference. Fig. 4 provides an overview of this scenario.

The scenario was evaluated on the testbed in real time using positions of the actual satellites propagated from their two-line elements. At 22:41:49 the spacecraft generated 40.5MB of emulated science data which was formed into a series of bundles of 6kB each. Reflecting the value of near-term delivery, a latency constraint of 20 minutes was attached to the bundles using the lifetime field in the bundle header. As shown in Fig. 5, the system sent a request for service several seconds later. This request message traversed the low-rate channel and

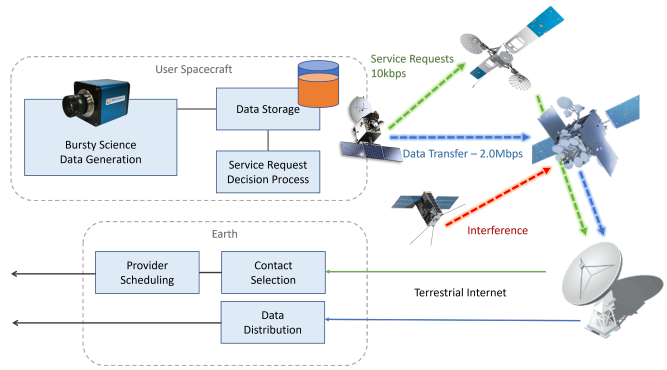

Fig. 4. Major components of scenario used for initial end-to-end test.

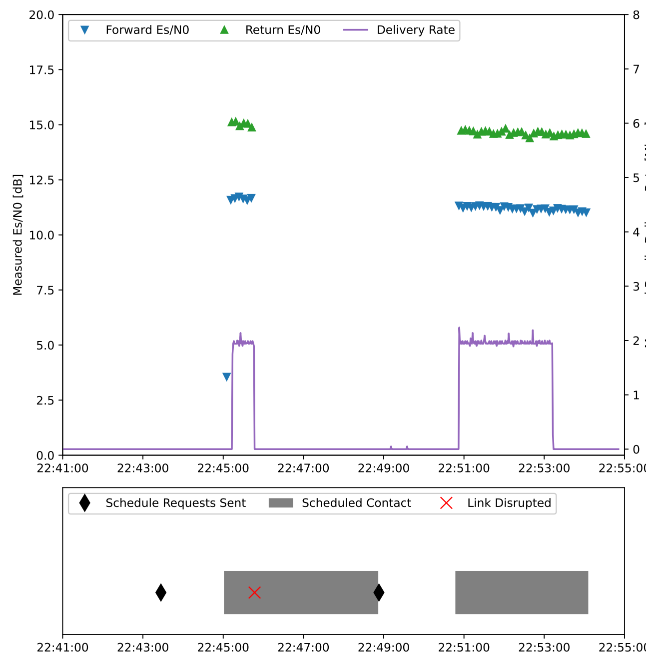

Fig. 5. Link quality and bundle delivery rate (top) during the two scheduled contacts (bottom). The red x at 22:45:46 indicates time of the link disruption.

was handled by the UIS server which communicated with the emulated Inmarsat scheduling system to reserve a contact from 22:45:02 - 22:48:51. The window was sized to transfer all data at the anticipated rate of 2Mbps plus an additional 60 seconds to account for signal acquisition and startup of bundle flow. A configuration message was also sent to the channel emulator (Section II-B) to emulate the physical link parameters for the duration of the contact.

Data transfer began shortly after contact start at the anticipated rate of 2Mbps. At 22:45:46, the link was disrupted. At that point 7.7MB had been successfully transferred with the remaining 32.8MB of data in onboard storage. Additionally, any bundles in transit during link disruption for which no ACK was received also remained in the spacecraft's storage. The system continued to retry the disrupted link by sending periodic pings for the duration of the contact. Immediately after the contact end the spacecraft reexamined its onboard

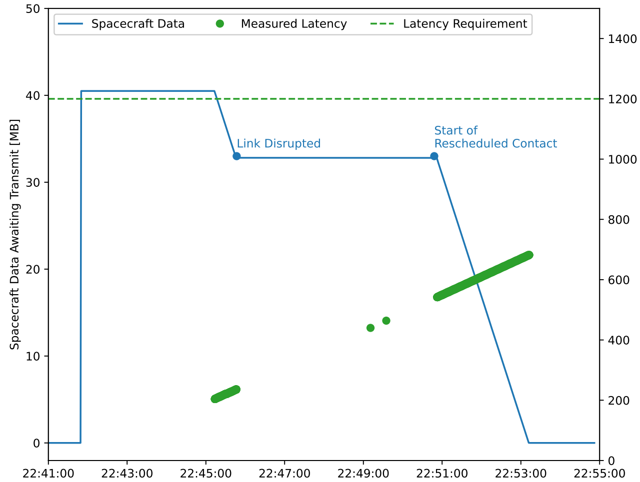

Fig. 6. Data queued for transmission onboard spacecraft and measured bundle latency over test duration. The dashed line represents the delivery requirement of 20 minutes from time of generation.

storage. Since the 32.8MB of data left to transmit was still on disk another request for service was issued. Following the same process this request was granted for 22:50:48 - 22:54:05 also over the emulated Inmarsat service.

Successful data flow began at the start of the rescheduled contact with the intermittent interference source now absent from the link. Fig. 6 shows the onboard storage volume decreasing to zero throughout the contact as all data is transferred. Latency is measured on a per bundle basis with full data completion shown to be 682 seconds after generation. This meets the 20 minute (1200 second) requirement. We note that Fig. 6 shows two individual bundles transferred over the low-rate control channel during periods where no high-rate channel was scheduled. This is due to a known issue with the router's handling of link disruption. In normal operations we instruct the router to ignore the low-rate channel and only route bundles over the high-rate scheduled channels. When the link goes down these several bundles are have already been routed and are queued for transmission. They are transferred when the radio is reconfigured to use the low-rate link. Since this is a usable link in our scenario these two bundles do not affect our final results.

V. Conclusion

As NASA transforms its space communication networks and continues to explore the Earth, Moon and beyond, emulation testbeds will provide a vital role in evaluating communication technologies. This paper presented an overview of a high-fidelity testbed focused on developing network-level optimizations for a dynamic and complex environment. The testbed enables evaluation in realistic operational environments, and provides a method to quantify performance metrics. Initial results were provided of a prototype automated system based on DTN and UIS running on the testbed. The testbed will continue to be developed and extended to support additional use cases related to multi-hop routing and multi-spacecraft constellations.

References

[1] M. Piasecki et al. "Development and demonstration of a wideband RF user terminal for roaming between Ka-band relay satellite networks". In: *38th International Communications Satellite Systems Conference (ICSSC 2021)*. Vol. 2021. 2021, pp. 241–248. DOI: 10.1049/icp.2022.0579.

[2] CesiumAstro Inc. *SDR-1001*. 2023. URL: https://www.cesiumastro.com/products/sdr-1001 (visited on 05/18/2023).

[3] Leigh Torgerson et al. *Delay-Tolerant Networking Architecture*. RFC 4838. Apr. 2007. DOI: 10.17487/RFC4838. URL: https://www.rfc-editor.org/info/rfc4838.

[4] David Israel et al. "Implementing Delay/Disruption Tolerant Networking for NASA's Plankton, Aerosol, Clouds, ocean Ecosystem (PACE) Mission". In: *The 16th International Conference on Space Operations (SpaceOps 2021)*. 2021.

[5] A. Schlesinger et al. "Delay/Disruption Tolerant Networking for the International Space Station (ISS)". In: *2017 IEEE Aerospace Conference*. 2017, pp. 1–14. DOI: 10.1109/AERO.2017.7943857.

[6] D3TN. "Field-tests for Demonstrating Low-Cost World-Wide LEO Communication Networks Leveraging DTN Protocols". In: *5th ESA CubeSat Industry Days*. 2021.

[7] NASA Glenn Research Center. *High-Rate Delay Tolerant Network*. URL: https://github.com/nasa/HDTN.

[8] D. Raible et al. *Developing High Performance Space Networking Capabilities for the International Space Station and Beyond*. Technical Memorandum 20220011407. NASA, 2022.

[9] K. Scott and S Burleigh. "RFC 5050, Bundle Protocol Specification". In: *IETF Network Working Group* (2007). URL: https://tools.ietf.org/html/rfc5050.

[10] M. Ramadas, S. Burleigh, and S. Farrell. "RFC 5326, Licklider Transmission Protocol - Specification ". In: *IETF Network Working Group* (2008). URL: https://datatracker.ietf.org/doc/html/rfc5326.

[11] K. E. Wilson et al. "Development of the Optical Communications Telescope Laboratory: A Laser Communications Relay Demonstration Ground Station". In: *IEEE International Conference on Services Oriented Computing (ICSOC 2012)*. 2012.

[12] J. Esper. *Draft LunaNet Interoperability Specification*. Technical Publication 20220004317. NASA, 2022.

[13] Adam Gannon et al. "Spacecraft-Initiated Scheduling of Commercial Communications Services". In: Oct. 2022. DOI: 10.1049/icp.2023.1394.

[14] A. Smith, E. Weston Brown, and F. Merat. "A Digital Testbed for Autonomous Spacecraft Communication Services". In: *2023 IEEE Space Hardware and Radio Conference*. IEEE. 2023, pp. 20–22.