SSC23-XII-08

Space Station Operations Capabilities in a Shoebox: Marshall Space Flight Center's Telescience Resource Kit

Jeff Lippincott, Michelle Schneider, Steve Chubb, Sam Digesu, Darren Wallace
NASA Marshall Space Flight Center
Mail Code: HP25 MSFC, AL 35812; 256-544-4260
jeff.lippincott@nasa.gov

ABSTRACT

The International Space Station (ISS) has provided the world an unprecedented capability, establishing a continuous human foothold in outer space for more than 22 years now. But maintaining that capability and supporting the ambitious portfolio of scientific research it hosts has required another unprecedented capability – providing ground-based operations support for the crew and a diverse manifest of research payloads, all simultaneously. To help meet this challenge, NASA's Marshall Space Flight Center (MSFC) developed TReK, the Telescience Resource Kit, providing a robust solution for data, command, metadata, and file transfer capabilities. In response to an increasingly wide and diverse need for operations support resources, the TReK team has worked to find ways to make the software more flexible in order to support a wide range of missions. Today it has supported not only hundreds of ISS payloads, but also free-flying spacecraft and even aircraft-based research. This presentation will discuss how the team has modified capabilities needed to enable 24/7/365 research aboard ISS to provide the flexibility needed to support Small Sat missions, going back to one of MSFC's first "minisatellite" missions in 2010 and beyond.

INTRODUCTION

The Huntsville Operations Support Center (HOSC) at Marshall Space Flight Center (MSFC) has provided ground systems and launch/on-orbit operations support for many NASA programs and projects since the 1950s. This includes support for a wide variety of spacecraft and payload operations environments including the Space Transportation System, Spacelab, the Hubble Space Telescope, Chandra, FASTSAT-HSV01, and the International Space Station (ISS). There have been many advances in technology over the last twenty years. With the longevity of the ISS program, there is a unique opportunity to see how those advancements changed the landscape and provided opportunities to expand capabilities for monitoring and controlling payloads.

In the early 1990's, the HOSC began preparing to transition from supporting the Spacelab program with missions lasting from 10 days to 3 weeks to supporting the ISS program operating 24 hours a day 7 days a week. During Spacelab payload developers travelled to MSFC to conduct their payload operations. This was not a viable solution for the ISS program, so the HOSC developed a suite of remote operations solutions to support monitoring and controlling payloads from anywhere in the world. The Telescience Resource Kit (TReK) software was one of the solutions developed to meet that need.

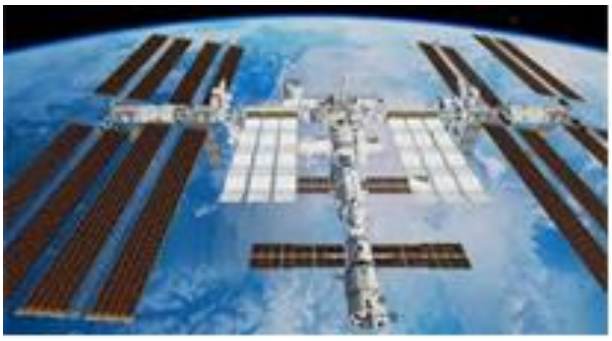

Figure 1 International Space Station

TReK is a suite of software applications and libraries that provide generic data system services. It provides local ground support system services and an interface to utilize remote services provided by the Huntsville Operations Support Center (HOSC). For the ISS Program, TReK provides ground system services for local and remote payload user sites including International Partners, Telescience Support Centers, and U.S. Investigators in locations worldwide. It can be used to monitor and control an asset onboard a spacecraft or on the ground. It is available for both Windows and Linux. TReK capabilities are provided through TReK applications and the TReK Application Programming Interface (API). The API provides the capability to write custom software for use onboard a spacecraft or on the ground utilizing just the capabilities needed for a particular operations concept. Software is

designed using libraries so each individual capability can be installed with a minimum amount of software. TReK can be used throughout the development lifecycle to support all activities including development in a laboratory environment, testing, simulations, and flight operations.

THE EARLY YEARS

As a manned space asset ISS has many levels of security requirements and unique interfaces that help protect the health and safety of both the crew and the spacecraft. With many payloads bound for ISS lasting from days to years it was not economically feasible to expect each payload team to build a ground system capable of communicating with ISS.

Since its inception in the late 1990s, TReK has been designed to run on low-cost commodity computers. The first operational release of TReK was on July 18, 2000 and was designed to run on Microsoft Windows NT® and subsequent Microsoft operating systems.

The early versions of TReK provide general telemetry services including receiving, processing, recording, forwarding, and playing back payload data. Parameters could be displayed in text, recorded in a file, or retrieved by user developed programs using the TReK provided ANSI-C Application Programming Interface (API).

Payload teams could also create, modify, send, track, and record commands. An ANSI-C API is also available for commanding that allows user developed programs to update the values of command parameters and send commands to the payload via the Payload Operations Center (POC) at MSFC.

The metadata that describes the layout of the telemetry and commands was stored in a Microsoft Access® database. This database could be built either from simple text-based files or from partial database downloads from the MSFC POC.

TReK also handled interfaces to various systems to receive telemetry and send commands. This included the ground systems at MSFC that are used for flight, simulations, and testing. Additionally, TReK provided standalone test support for the Payload Rack Checkout Unit (PRCU), Suitcase Simulator (ScS), and other systems designed for payload integration testing with the onboard ISS systems prior to flight.

These early capabilities still exist today and have been enhanced based on the lessons learned from the ISS program and other projects. The following paragraphs describe the evolution of TReK based on the lessons learned and a glimpse into what the future holds.

TELEMETRY

A key element in operating any payload or spacecraft is insight into its current state. Telemetry data is traditionally downlinked as CCSDS packets and processed on the ground to parameter level data that can be monitored for the health and safety of the payload.

Packet Identification and Attributes

The early versions of TReK determined how to process incoming telemetry data using fixed identification schemes that were defined in various ISS program documentation. While this worked well for ISS payload data processing, it proved to be difficult to add new identification schemes quickly since some code changes were required.

TReK introduced a generic form of identification called *packet templates* to ease the burden of adding new types of data. Internally, TReK creates a *packet header processor* from the *packet template*. The *packet header processor* looks at incoming data and checks the values in the packet to determine if it meets the criteria for processing. Each incoming packet is assigned a *packet key* which is a globally unique name that can be used as a reference to the packet.

Packet templates have already proven useful for ISS payload teams who needed an expanded definition for packet identification to meet the unique processing needs for their payload.

In addition to the flexibility of having one or more identifiers for a packet, there are four other generic attributes associated with *packet templates*: length, counter, time stamp, and checksum.

The *length* of a packet can include an offset to determine the actual number of bytes in the packet. This is important for any transport protocol such as TCP which may require packets to be reassembled as they are received.

The *counter* in a packet allows TReK to monitor incoming stream data to detect any out of sequence or missing data in the stream.

The *time stamp* in a packet can be any of the many time data types available in TReK. The time stamp is used when recording packets to provide reference to the time the packet was generated at the source. The *time stamp* and *counter* are both used to reorder and remove duplicates for any packet stream.

The *checksum* in a packet can be used to detect any bit errors that may have occurred during the transmission of the packet.

Recording

All versions of TReK have provided packet recording capabilities which include the ability to close record files based on size and time. The earliest versions of TReK only allowed recording to be set up for an incoming identified packet going to a single record file.

The newest versions of TReK have the additional capability of recording multiple packets with the same identification scheme to a single file. The files can also be automatically segregated to separate record directories based on the *packet key* used for identification. A third option to allow complete user control of filenames and directories for each identified packet is available.

The newest versions of TReK can also perform real-time reordering and merging of packet data including removing any duplicate data.

Processing

TReK has always provided many numeric, string, and time data types that can be converted to the local computer's internal representation of the data. Additionally, TReK can provide further processing and monitoring of the data.

TReK originally allowed two levels of alarms: caution and warning, but this has been expanded to five levels with each level capable of having a project specific name for the level.

All versions of TReK have provided polynomial and point pair (spline) calibration, but recently two new types of calibration have been added that allow more detailed processing: user calibration and math calibration.

User calibration allows you to use a template class and write your own calibration code. TReK can import your custom code into the TReK processing architecture giving you complete control over how calibration is performed for any given parameter.

Math calibration also provides flexibility without the need to write any code. Math calibration uses the Reverse Polish Notation (RPN) to allow you to create mathematical equations for the calibration of parameter data. Math calibration also allows you to use the value of other parameters in the calibration calculation as needed.

Application Programming Interface (API)

One of the key components of TReK is the ability for the user community to leverage TReK data processing into custom programs using an API. The first version of the API was completely ANSI-C based allowing integration into multiple programming languages and tools. TReK also provided a wrapper to the .NET Framework of all the telemetry APIs allowing for easier integration into C# and other .NET Framework programming languages.

The newest versions of TReK have a base API that is C++. This provides for much more details and control over processing of packet and parameter level data. Even though TReK uses the C++ API, the ANSI-C and .NET Framework versions are still completely supported. Many payload teams still have ANSI-C based user applications that were written over 20 years ago and that work seamlessly with newer versions of TReK.

TReK has recently expanded the supported telemetry APIs to include a Python wrapper of all the C++ telemetry APIs. This allows programs written for Python to easily incorporate telemetry processing capabilities.

Data Generation

Payload teams often need to generate telemetry data to test their ground system applications. Part of testing the ground systems is to see how they handle unexpected data. TReK provides a data generation capability that can simulate any packet that TReK processes. This capability supports insertion of nominal and error data in a repeatable manner to aid in payload testing.

COMMANDING

Another core component of operations is the ability to control the payload through commanding.

Packet Templates

TReK uses the same concept of packet templates for generating command packets that is used for processing telemetry data. For commanding the packet templates provide structure to ensure packets are automatically populated with the correct values for each attribute in the outgoing packet.

Interface Abstraction

Even within the ISS program there are a variety of command interfaces used for flight and testing. TReK uses interface abstraction to allow easier integration of new command destinations into the code base. The functionality provided by each interface can vary

greatly. Some interfaces provide multiple status messages describing the state of the command system along with responses to each command indicating success or failure at various stages of the uplink path. Other interfaces provide no feedback for the commands that are sent. No matter what command system is receiving the command, the TReK system handles it seamlessly.

Application Programming Interface

Just like the telemetry APIs, the earliest versions of TReK provided an ANSI-C interface for commanding as well as a .NET Framework wrapper. The command API allowed user programs to update command values as well as send the command to a designated destination. The API abstracts the sending of commands so that as the destination changes from testing to flight the user program does not have to be updated.

The command API allows commands to be built in multiple ways. The first method is to build the contents of the command based on the metadata definitions associated with the command destination. The second method to build commands allows a user program to provide the actual bit pattern that is to be uplinked. If supported by the destination, the command API can also request that the destination build the command uplink pattern based on its current metadata contents.

The command API in the newest versions of TReK is now C++ based with complete backward capability for the ANSI-C API and .NET Framework wrapper.

The Command Bridge

Testing for ISS payloads occurs throughout the life cycle of the payload. Unfortunately, the different command interfaces prevented the ground systems used by the HOSC from being used as part of the testing. This meant that the payload team was unable to test with flight-like ground interfaces and flight-like onboard interfaces simultaneously.

The TReK abstraction of interfaces proved useful to solve this issue. The Command Bridge was created to allow the HOSC ground systems to send a command over the network to a TReK system which translated the command to any of the test systems for ISS.

This capability was further abstracted to allow any TReK command interface to allow bridging. Some payload teams have taken advantage of this capability by integrating legacy or other command and control systems into the ISS world at minimal cost.

METADATA

Telemetry and commanding are the core of most interactions with payloads onboard ISS. To use these functions, you need a robust set of metadata that describes the attributes of the parameters and packets that make up the telemetry and commands. TReK provides several different types of capabilities to make metadata creation simple in the ISS world.

Since ISS is a large program all the payload metadata is added to a single database along with metadata for onboard systems for management at the program level. Individual payload teams don't need access to the entire data set. The ISS program provides a subset of information to payload teams in a series of delimited text files that are referred to as a partial database download.

The early versions of TReK converted the information provided by the partial database download into a Microsoft Access® database. The TReK telemetry and command applications and APIs were able to pull the information needed for processing from the Microsoft Database®. Since the newest versions of TReK also support RedHat Linux, the import of this metadata is now into SQLite databases.

Payload teams often need to access other telemetry provided by the ISS core systems such as the location of the spacecraft or the beta angle. Since the information needed by a single payload is often spread across multiple data streams the ground processing software at the MSFC POC can provide custom data packets that only contain the requested parameters. The metadata associated with the custom packets are provided to the payload teams as a delimited text file. This text file can be imported into TReK for use by the telemetry processing software.

While relational databases provide a good way to organize large sets of information, TReK also has a need to describe the individual commands and telemetry items in a standalone environment. The metadata that is stored in the SQLite database can be exported into XML files that are used by TReK programs and APIs that need command and telemetry access.

TReK provides a graphical user interface to pull all the metadata features together. In addition to importing information from external sources, the Metadata application provides a simple drag and drop environment to create telemetry and commands and their associated attributes such as calibration.

Often payload software will use a structure of data to format the telemetry or command data so it can be easily accessed in the flight software. TReK can import C data structures to create metadata definitions that can be used as telemetry or commands. Another feature of the import of the C-structure is the ability to assign attributes to parameter data by using C comments. For example, if a parameter has a valid range, it can be specified with special comment tags.

As the ISS program gained experience in integrating many payloads it became apparently that a new metadata format was needed. Originally all payload metadata had to be manually entered into a system named the Payload Data Library (PDL). Each telemetry and command data item had to be entered through PDL. While this data could be exported back to the payload teams, there was no mechanism for the payload team to provide the metadata in a format for PDL to import.

The Command Data Exchange Format (CDEF) was created to solve this problem. CDEF is defined by an XML schema and incorporates all the telemetry and command attributes that are used in the ISS program databases. TReK can import metadata from CDEF, as well as export metadata to CDEF for import into other ISS systems. This is beneficial to ISS payload teams as often early development and testing of the payload occurs prior to the time required for entry of the metadata into PDL. This allows payload teams to use the many features available to create metadata within TReK and export that information instead of having to enter it again with the possibility of introducing errors.

ENCRYPTION

Providing secure access to payload data is critical to the ISS program. Security requirements are continually evolving, and the ISS program identified a need for payloads to encrypt files and data with FIPS 140-2 compliance for any network transfer of the data.

TReK provides an encryption library to secure transmissions of proprietary data between flight and ground eliminating the need for the development of independent encryption schemes by different ISS payload teams. TReK uses OpenSSL's FIPS 140-2 validated cryptographic module and public/private key pairs to encrypt and decrypt data. TReK encryption library support is provided on 32-bit and 64-bit Linux operating systems and 64-bit Windows operating systems. Both the flight platform and ground platform must use the TReK encryption library to encrypt and decrypt data.

TReK's public key/private key encryption architecture is based on Elliptic Curve Cryptography (ECC) using curve P-256 providing 128-bit security with 128 or 256-bit keys. The cipher packages included with the TReK encryption library are the Advance Encryption Standard (AES) Galois/Counter Mode (AES GCM) and the AES Counter with CBC-MAC (AES CCM) ciphers offering confidentiality, authenticity, and integrity. The library supports 128 and 256-bit cipher key sizes and provides AES 128 and 256-bit key-wrap/unwrap functions. Fresh Cipher Encryption Keys (CEK) are created for data using a Password-Based Key Derivation Function 2 (PBKDF2). The TReK encryption library generates a new CEK for every encrypted file and may be configured to generate a new CEK for every encrypted packet in a packet stream. The TReK encryption library may also be configured to generate a new CEK, for a packet stream, once every "x" seconds to support encryption of high-rate packet stream. No "encryption handshaking" is required between flight and ground hardware during the encryption and decryption of packets.

FILE TRANSFER

Transferring files between space and ground assets is an operational requirement for most ISS payloads and small satellites. When the first payloads began operating on ISS there were no file transfer standards to follow so each payload team developed custom solutions using traditional telemetry and commands. This inefficiency was addressed when CCSDS published the CCSDS File Delivery Protocol (CFDP) Blue Book. The TReK team was fortunate to acquire a CFDP library developed by Goddard Space Flight Center (GSFC) that implements a CFDP “put” file directive as defined in the standard. The GSFC CFDP library was enhanced by TReK to operate on the Windows operating system as well as the original Unix/Linux implementation. TReK also updated the GSFC library to include support for file store actions (e.g., delete file, create directory), messages, and “gets” implemented as “proxy puts” as defined in the CFDP standard. These enhancements were delivered to GSFC as part of the agreement to use their library. The TReK implementation of CFDP (referred internally as native CFDP) includes class 1 service, a “send and forget” level of service that sends the files without any acknowledgement of their receipt and class 2 service which ensures file delivery by requiring positive acknowledgements and requests for retransmission of missing data from the recipient.

The configuration of CFDP requires a lot of guesswork and testing to settle on proper numbers for a project for retransmission attempts and timeouts. TReK has implemented a simple four-byte message that is sent

outside of the CFDP protocol that allows TReK implementations to automatically detect when a two-way communication path is available. When the communication path is no longer available, TReK can automatically suspend all CFDP transactions. By suspending the CFDP transactions the configuration of CFDP become much simpler. Once the communication path is available again, TReK automatically resumes the CFDP transactions allowing file transfers to complete.

Drop Boxes

Another enhancement provided by the TReK CFDP implementation is the concept of drop boxes. A drop box is a directory on the local system (flight or ground) that is tied to a particular action. The drop box will have a corresponding location on the remote system (ground or flight). There are several types of drop boxes available in TReK and these drop boxes can be chained together to provide all the capabilities in a single file transfer.

The primary drop box is the CFDP drop box. This allows files to be transferred to a corresponding directory on the remote system by simply writing a file to the specified directory. The TReK CFDP implementations monitor for files in the directory that are no longer opened by another program and transfers the file with a CFDP "put" directive. Once the file is confirmed to be successfully transferred, TReK can be configured to either delete the file or move it to an archive location.

The transfer of multi-Gigabyte files is difficult over communication links with limited transmission bandwidths and the potential for dropping packets. Consequently, multi-Gigabyte files transfers often fail because the recipient exceeds the number of retransmission requests allowed by the CFDP configuration or the retransmission request packet identifying the missing file segments is so large and fragmented it fails to reach its destination and is never processed. TReK CFDP provides a fragmentation drop box that can split a large file into smaller pieces of a configurable size. When chained to a CFDP drop box, these smaller files are transmitted to the remote system. The remote system can then use a defragmentation drop box to put the pieces back to the original multi-Gigabyte file. To minimize the possibility of dropped retransmission requests, the TReK native CFDP library may be configured to automatically resize retransmission requests packets that become too large due to many CFDP PDU packets getting dropped during the file transfer transaction.

The content of files can be protected by using the TReK encryption and decryption drop boxes or native CFDP can be configured to encrypt and decrypt the CFDP packet streams associated with CFDP transactions.

The drop box features in TReK are used extensively by ISS payloads to provide a simple interface for moving large amounts of data seamlessly. A payload that generates files can use CFDP to transfer the files to the ground system through simple configuration files.

Extra Security

When sending a file that may contain commands it is important to prevent a nefarious attack on the destination via a playback of data even if that data is encrypted. The TReK native CFDP implementation can be configured to encrypt a time stamp, time to live value, and sequence count to provide replay resistant time authentication.

Setting Up CFDP

TReK has both a graphical user interface and console-based applications for CFDP. The graphical user interface allows for the easy configuration of the capabilities provided by CFDP. It allows simple interaction with the remote side of the interface with drag and drop transfer of files and complete statistics to keep up with the transactions in real-time. The console-based version is ideal for systems where a graphical user interface isn't available or when you want to run CFDP in the background. The configuration file used by the graphical user interface is 100% compatible with the console and can be used to configure the console program.

DELAY TOLERANT NETWORK (DTN)

Even though ISS has much greater communication coverage than most spacecraft, it is still subject to interruptions in communication and some propagation delays during routine operations. The ISS program has implemented Delay Tolerant Networking, also called Disruption Tolerant Networking, to ease the burden caused by the loss of the communication path. DTN consists of multiple protocols defined in CCSDS blue books. These protocols handle the storage and retransmission of data and include mechanisms to ensure the data is received by the ultimate destination.

TReK uses the Jet Propulsion Laboratory (JPL) implementation of the DTN protocols called the Interplanetary Overlay Network (ION). The fundamental building block of DTN is the Bundle Protocol (BP). A bundle is a unit of data transmitted between nodes in DTN. TReK uses BP for two primary purposes: file transfers and telemetry.

File Transfer

When the CFDP applications in TReK use BP as the transport mechanism it is referred to as "ION CFDP". Since the reliability of data transmission is provided by the DTN protocols, ION CFDP only implements the class 1 "put" directive along with the file storages directives and messages.

Both the graphical user interface and console-based versions of the CFDP applications in TReK support ION CFDP. All the additional capabilities provide in CFDP such as encryption and the various dropbox types are available when using ION CFDP as well. It is very simple to change the configuration to use either Native CFDP or ION CFDP.

Since the ION CFDP implementation uses class 1, there is no final CFDP transaction result to know when and if the transfer is completed successfully. TReK has implemented a CFDP message to allow knowledge of the transaction result in all CFDP applications. Additionally, TReK has implemented a "get" directive for ION CFDP using CFDP messaging. While both these additions are specific to TReK and require TReK at both endpoints for use, they are implemented within the CFDP standard and will not break interoperability.

Telemetry

TReK also uses BP for the receipt of telemetry. Instead of configuring for incoming data over an IP based network, you simply provide equivalent information for retrieval through DTN. All the capabilities described earlier for telemetry data are available for telemetry received through DTN. It is also possible to forward data with BP through DTN.

ION Setup

The DTN protocols implemented in ION require quite a bit of setup and can be very difficult and error prone for new users of ION. TReK provides a graphical user interface with drag and drop capabilities to define the DTN network and generate the configuration files needed for ION as well as scripts that will start all the needed ION applications.

Since there are well know DTN configuration paths available for ISS, templates of the networks are made available to users to simplify the configuration setup. The typical ISS user only needs to edit one of the templates and make payload specific changes such as the data storage needed or the data rates the payload should maintain prior to generating the configuration files for both the payload computer and ground system.

EMAIL AND TEXT

Minimizing personnel time on ground consoles is a key factor in reducing the operation cost of a project. Providing tools to that allow teams to minimize console time has been a primary goal for TReK as it has evolved over the years. TReK now has an automated email and text capability that uses the cURL open-source software library, to log into and send the texts, email messages and email file attachments to a designated group of one or more recipients by communicating with a Simple Mail Transport Protocol (SMTP) email server. The email and text capability is integrated into multiple TReK applications which allows timely notification of operations personnel even when away from the console. Examples of this feature in TReK includes the sending of email and text when a parameter has limit violations, a command connection is lost, successful or failed files transfers, and a summary report for file transactions every 12 or 24 hours. The TReK email library also includes the ability to create a drop box that sends a file as a file attachment or sends the content of the file as a text or email message.

EXTENDING CAPABILITIES WITH APIS

TReK has always provided APIs to access telemetry and command functions from user developed programs. These APIs allows payload teams to provide custom applications to their operators to better meet the needs of the payload.

As the TReK design evolves to a more decentralized architecture, each feature in TReK is evaluated to see if an opportunity exists to provide some of the features typically available in the core TReK applications as APIs that could be integrated into user programs.

The CFDP API allows users to integrate file transfer functions directly into their program. These file transfer functions work with both Native CFDP and ION CFDP. The CFDP API initializes from the same configuration file format used by both the CFDP GUI and the CFDP Console applications in TReK and provides access to the full range of CFDP capabilities.

The Device API provides access to functions for both IP and BP communications abstracted to run on the Windows and Linux operating systems. The power of the packet templates can be employed to identify incoming data and callbacks are available to provide the identified packets to a user implemented function.

The Record API can be used to record any data received on an IP socket. The generated files have all the file naming capabilities found in the TReK applications as well as the ability to close files based on

time or file size. Additionally, functions are provided to write directly to the file from a user program without any network communication.

The Email and Text API is available to allow user programs to integrate notification features into custom programs. Once configured a simple function call is all that is needed to send an email and/or text to one or more recipients to provide payload specific notifications.

The core TReK code is written in C++ and the telemetry and command APIs have C++ classes available for processing. Those APIs also have ANSI-C wrappers to allow for integration into programming environments that do not support object-oriented programming. There are also .NET framework and Python wrappers available for the telemetry and command APIs.

All the other APIs in TReK are ANSI-C to allow for integration into user programs. TReK is evaluating adding Python and .NET Core wrappers for the various APIs in TReK to make it even easier to integrated functionality into different programming environments.

ONBOARD SUPPORT

Many of the capabilities in TReK also can be used on the payload computer. The most popular of these capabilities is CFDP. Using CFDP on payloads to transfer files to the ground simplifies the onboard software design by eliminating the need for the payload to write file transfer software.

Another popular option for onboard use of TReK is the DTN configuration setup. DTN allows data to be transferred to the ground without the need for a user to be on console. Since the retransmission of data is handled by the core ISS systems in the store and forward network, the files are waiting for retrieval by operations personnel when they are once again on console.

Many of the onboard ISS systems that connect to payloads have specific interface requirements that each payload must meet. TReK has written API abstractions for these interfaces to make integration into ISS easier. Instead of each payload team recreating the code necessary for communicating with onboard systems, they can make API function calls to receive commands and send telemetry as well as provide the necessary status required by the onboard systems.

Many payload teams use different CPU architectures in their flight computers. TReK provides a service to cross-compile the code needed for flight to different computer architectures upon request. This allows payload teams to leverage TReK capabilities on multiple computing platform.

SUPPORTING OTHER PROGRAMS

TReK has also supported programs outside of the ISS payload world. TReK was used to support development, testing, and flight operations of the Fast Affordable Science and Technology Satellite (FASTSAT). NASA/MSFC in collaboration with the Department of Defense (DoD) Space Test Program (STP) and industry partners developed the spacecraft to demonstrate a fast, affordable, and low-cost bus technology and mission integration approach model. The FASTSAT-HSV01 satellite carried six small payloads including three technology demonstration experiments and three atmospheric research instruments. TReK provided ground system services on FASTSAT Mission Operations Workstations and at Remote Sites.

Figure 2 FASTSAT-HSV01

TReK was used to support the WB57 Ascent Vehicle Experiment (WAVE). WAVE recorded High Definition (HD) video and Near Infrared (NIR) images of the Space Shuttle launch. The goal was to improve the detection and analysis of anomalies (e.g., insulation loss, tile loss) during launch and landing. NIR and HD Video Cameras were mounted on a gimbaled optical bench in the nose of NASA's WB57 Canberra aircraft. TReK was installed on a computer in the aircraft cockpit and used to record the NIR images. A WAVE version of the TReK software was loaded on a flight qualified Crystal CS500 computer to control the

acquisition, time tagging and recording of the NIR images. TReK also received commands from and sent command responses to the flight control system via an RS232 interface. TReK controlled the camera through an Ethernet interface and time tagged the images using a Masterclock Time Code Reader (TCR) card receiving an IRIG B timing signal. The images below show the WB57 aircraft, camera, and Crystal computer.

Figure 3 WB57, Infrared Camera, and Crystal CS500 Computer

INTO THE FUTURE

TReK continues to evolve to support more large projects and small satellites. Updates are currently being made to support the Gateway program in a similar manner as ISS support over the last 20 plus years. Many of the abstractions made to support ISS have proven beneficial for Gateway. The packet identification for Gateway includes more than half a dozen different fields and the packet template has made integration of the complex scheme very simple.

TReK is also adding support for metadata defined in the CCSDS XML Telemetric and Command Exchange (XTCE) format. Incorporation of this standard will provide even more interoperability with other systems that produce and consume XTCE metadata.

To further support small satellite missions TReK is part of an automation demonstration at MSFC which includes automatic connection to the Deep Space Network (DSN) and other networks capable of reaching satellites through the HOSC.

TReK is adding a generic command sequence tool that will allow generation of command timeline files for autonomous operations. The generic tool is capable of writing project specific files for use on spacecraft as well as files that can be used on the ground. TReK can read and operate according to the ground timeline and create *significant event* notifications that are automatically handled according to project needs.

CONCLUSION

TReK is a comprehensive software solution for vehicle/payload operations commanding, monitoring, and payload activities. It allows users to monitor and control assets in space or on the ground. Plans are in place to continue to evolve TReK with automation and remote operations as its core capabilities to support local and remote manned and automated payload and small satellite operations.