A Cloud-Based, Open-Source, Command-and-Control Software Paradigm for Space Situational Awareness

Ryan Melton and Jason Thomas
Ball Aerospace, 1600 Commerce St, Boulder, CO 80301

1. ABSTRACT

With the rapid growth in the number of space actors, there has been a marked increase in the complexity and diversity of software systems utilized to support SSA target tracking, indication, warning, and collision avoidance. Historically, most SSA software has been constructed with "closed" proprietary code, which limits interoperability, inhibits the code transparency that some SSA customers need to develop domain expertise, and prevents the rapid injection of innovative concepts into these systems. Open-source aerospace software, a rapidly emerging, alternative trend in code development, is based on open collaboration, which has the potential to bring greater transparency, interoperability, flexibility, and reduced development costs. Open-source software is easily adaptable, geared to rapidly changing mission needs, and can generally be delivered at lower costs to meet mission requirements.

This paper outlines Ball's COSMOS C2 system, a fully open-source, web-enabled, command-and-control software architecture which provides several unique capabilities to move the current legacy SSA software paradigm to an open source model that effectively enables pre- and post-launch asset command and control. Among the unique characteristics of COSMOS is the ease with which it can integrate with diverse hardware. This characteristic enables COSMOS to serve as the command-and-control platform for the full life-cycle development of SSA assets, from board test, to box test, to system integration and test, to on-orbit operations. The use of a modern scripting language, Ruby, also permits automated procedures to provide highly complex decision making for the tasking of SSA assets based on both telemetry data and data received from outside sources. Detailed logging enables quick anomaly detection and resolution. Integrated real-time and offline data graphing renders the visualization of the both ground and on-orbit assets simple and straightforward.

2. COSMOS BENEFITS

The COSMOS C2 software package provides many key benefits to the SSA ground system architect and operator. The open source nature of COSMOS means it is completely free to start using COSMOS. If additional features or functionality is needed, the source code is freely available for inspection and modification. The open source nature of COSMOS means more and more individuals and companies are being exposed to COSMOS resulting in increased mindshare among operators and developers.

COSMOS can quickly interface with many kinds of targets. Any embedded system that provides a communication interface can be connected to COSMOS. COSMOS ships with interfaces for connecting over TCP/IP, UDP, and serial connections but also supports custom interfaces to connect to anything that a computer can talk to.

All the COSMOS tools are configured with plain text configuration files. This includes the standard COSMOS interfaces and the target command and telemetry definitions. This makes configuration easy by allowing copy and paste as well as more complex templating. It also enables standard configuration management tools which allow text based diffs and searchable history.

COSMOS has a rich API which allows independent tools to interact with the command and telemetry stream. The COSMOS API makes sending commands and checking telemetry easy. However, you are not constrained by your scripting language. COSMOS scripts are written in Ruby, a modern, fully functional scripting language. This allows you to read and write files, and perform live processing that most other systems force you to run offline. An official Python API is supported as well as the native Ruby API. COSMOS is fully cross platform and utilities the QT GUI framework to operate seamlessly on Windows, Linux and Mac OS X.

COSMOS has a rich history in test and can be used at all levels of integration from board level test, box level test, payload integration and test, to spacecraft integration and test. This makes COSMOS an excellent choice to provide

a consistent user interface throughout the full lifecycle of a product. It means you can utilize the same procedures and scripts in operations that were created during development.

Everything in the COSMOS C2 System is logged, and even more importantly, tools are provided to easily interpret and use the logs. Whenever an anomaly occurs there are tools already written that are ready to dig into the logs and help figure out what happened. Data can be visualized with telemetry displays and graphs both in real-time and via log files.

2. COSMOS TOOLS & ARCHITECTURE

Ball Aerospace COSMOS comes with the following set of 17 applications that are directly available for use with minimal to no configuration.

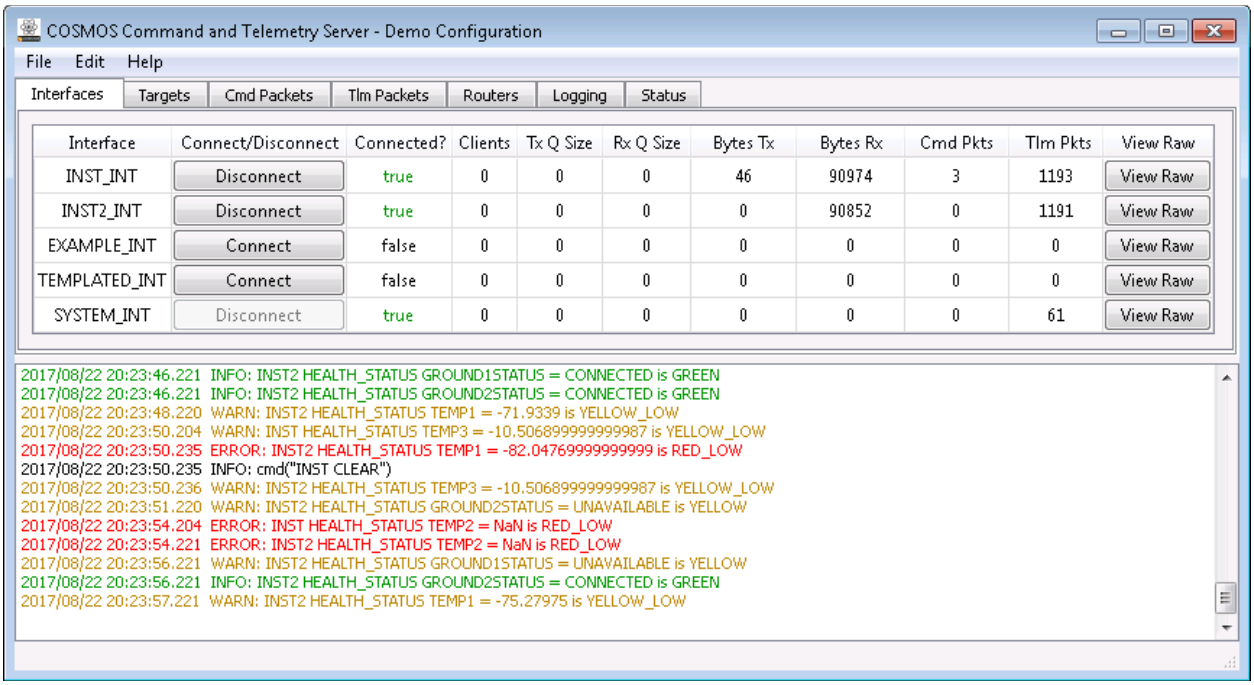

Fig 1. Command and Telemetry Server

The Command and Telemetry Server acts as the hub of the real-time portion of COSMOS. All commands and telemetry packets pass through this tool ensuring everything that happens is logged. It provides real-time commanding, telemetry reception, logging, limits monitoring, packet routing, and system status.

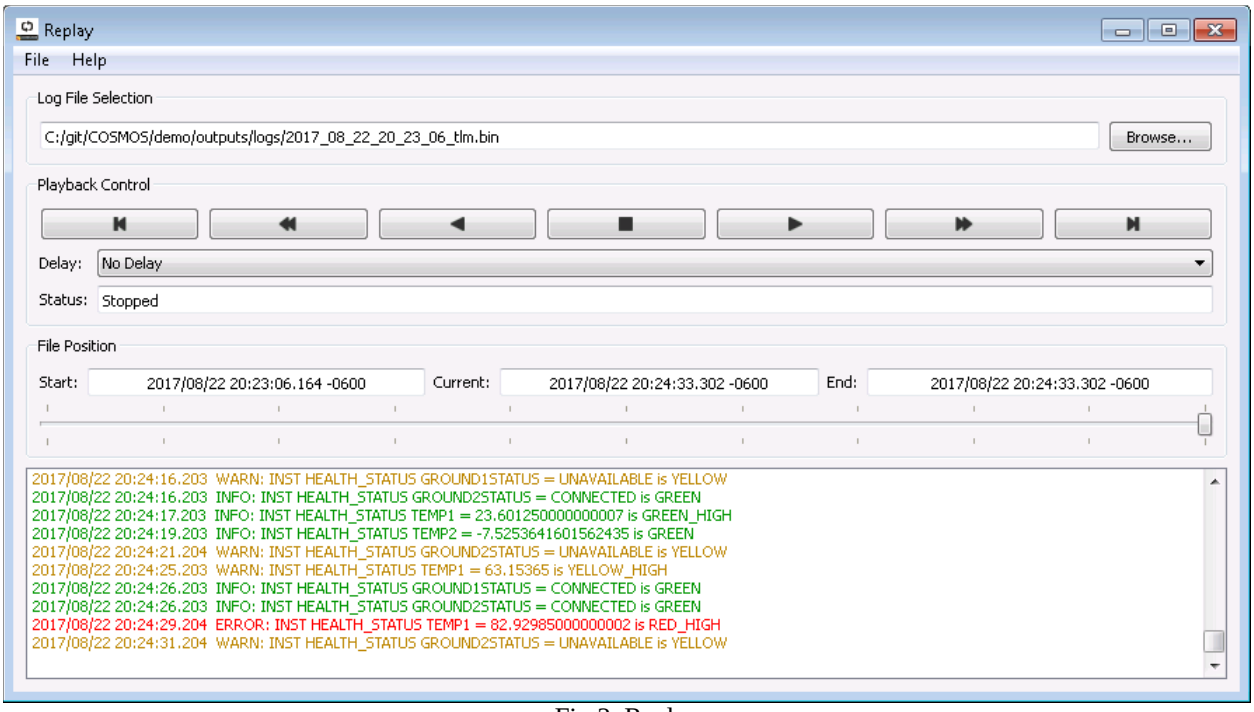

Fig 2. Replay

Replay simulates the Command and Telemetry Server for telemetry packet log file playback. This enables use of any of the real-time tools with logged data. Replay is great for playing back scenarios and viewing them on telemetry screens.

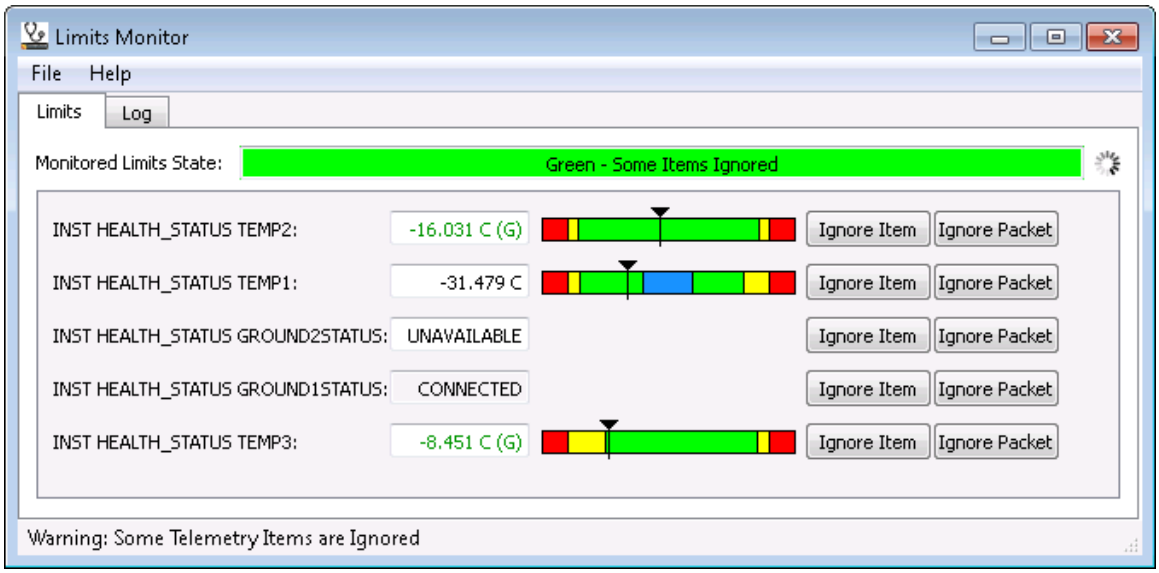

Fig 3. Limits Monitor

Limits Monitor monitors telemetry with defined limits and shows items that are currently out of limits or have violated limits since the tool was started. Expected violations can be easily ignored.

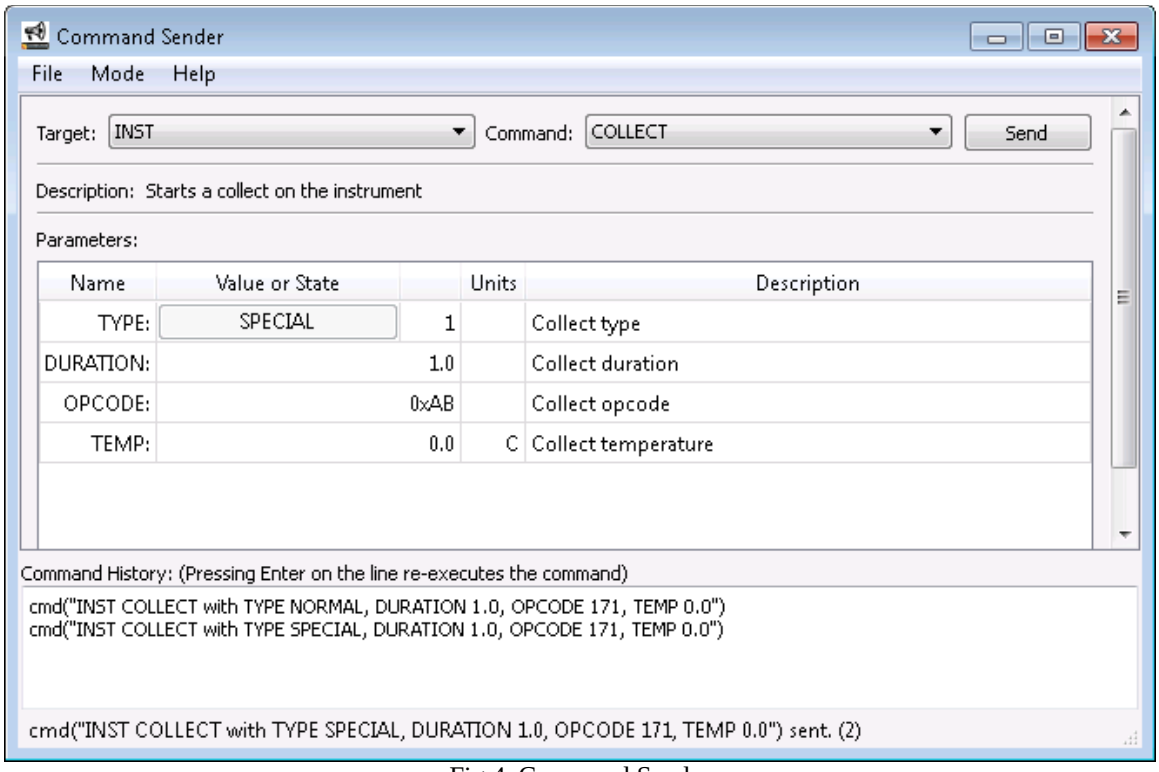

Fig 4. Command Sender

Command Sender provides a graphical interface for manually sending individual commands. Drop down selection of every command and command parameter in the system makes sending individual commands easy. A history pane makes resending previous commands easy.

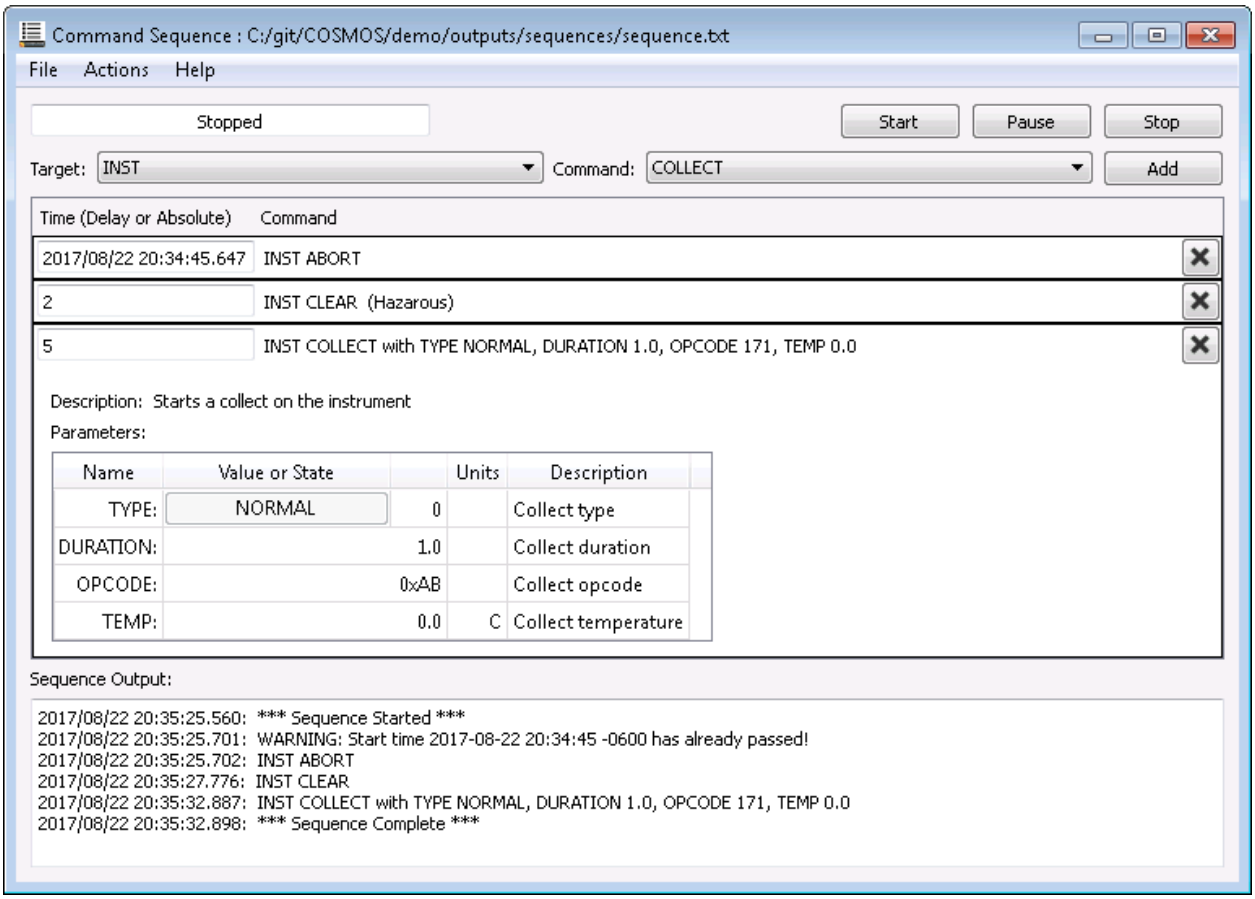

Fig 5. Command Sequence

Command Sequence allows for creating a sequence of commands that can be executed as agroup. Commands can be absolute time tagged or have relative delays between commands. The full GUI capability of Command Sendor is available to customize indivual commands.

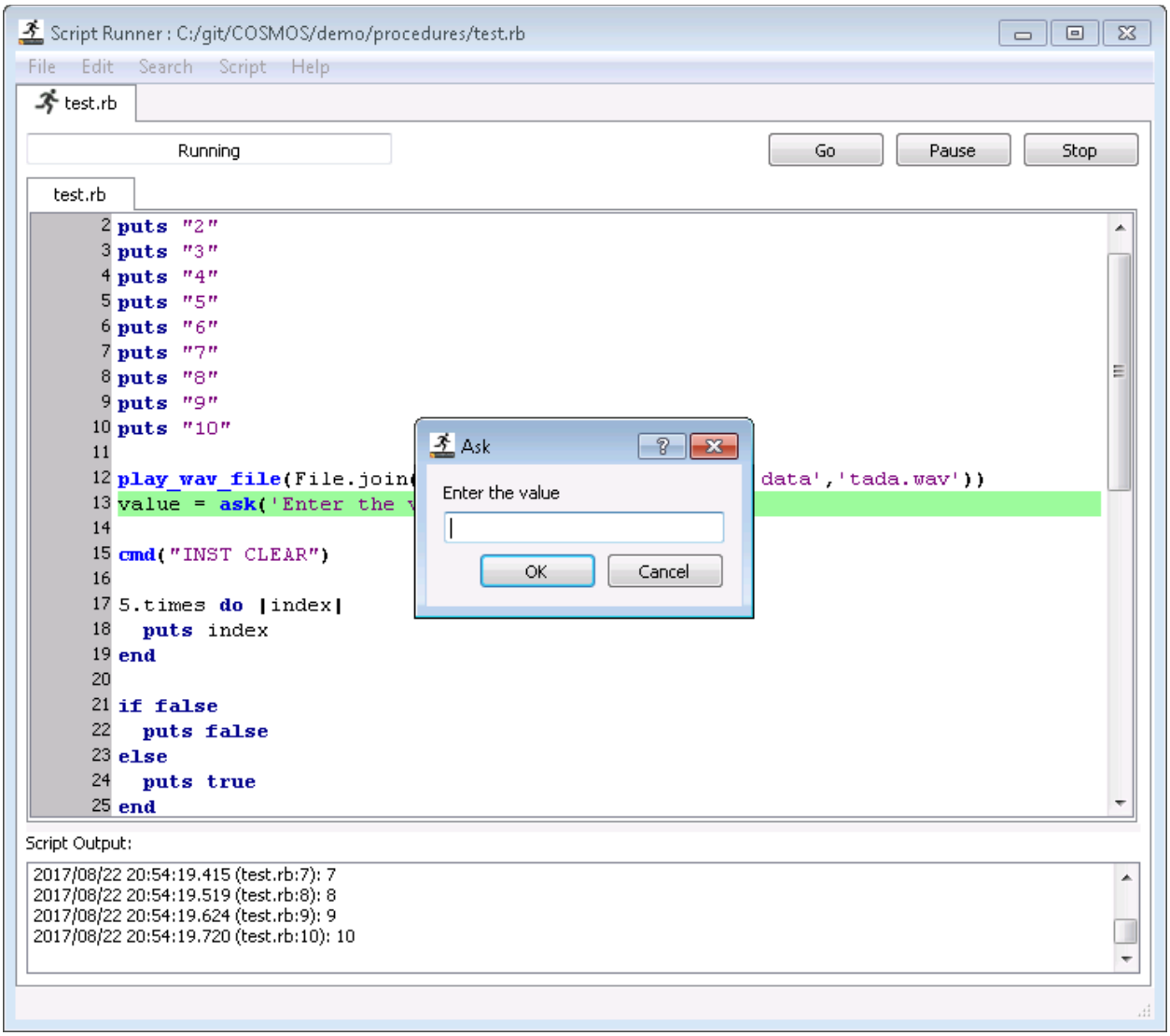

Fig 6. Script Runner

Script Runner executes test scripts and provides highlighting of the currently executing line. Scripts pause if any error occurs, breakpoints can be added, and lines can be re-executed after a problem has been corrected.

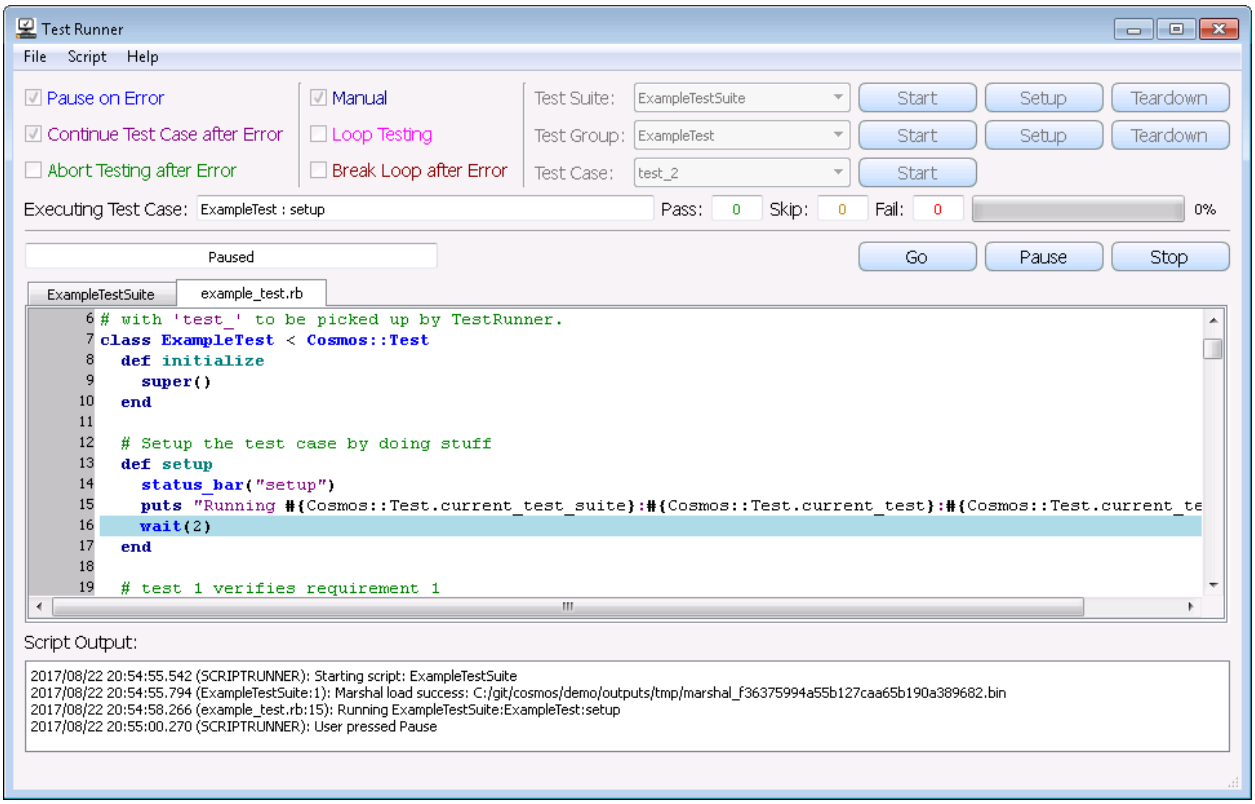

Fig 7. Test Runner

Test Runner provides a high-level framework for system level testing including automatic test report generation. Test Runner brings the best features of software unit level testing to system level integration and test by breaking tests down into easy understandable test cases. Users can execute entire test procedures or just the specific test cases they need to run for integration or regression tests.

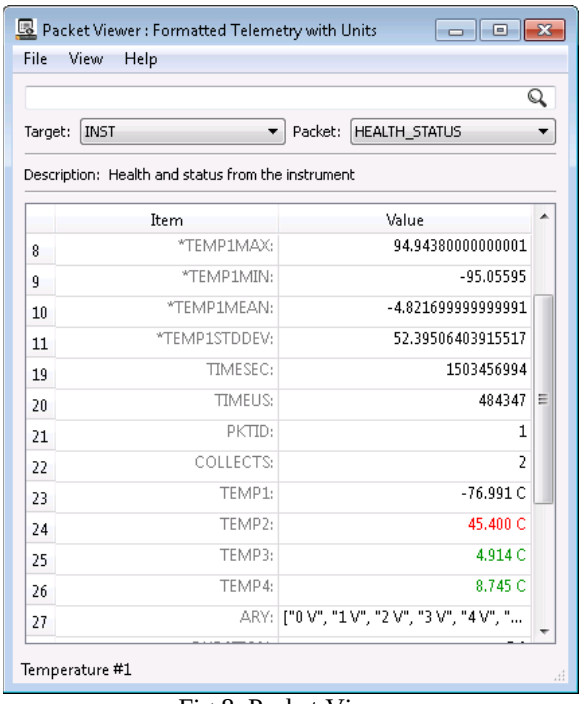

Fig 8. Packet Viewer

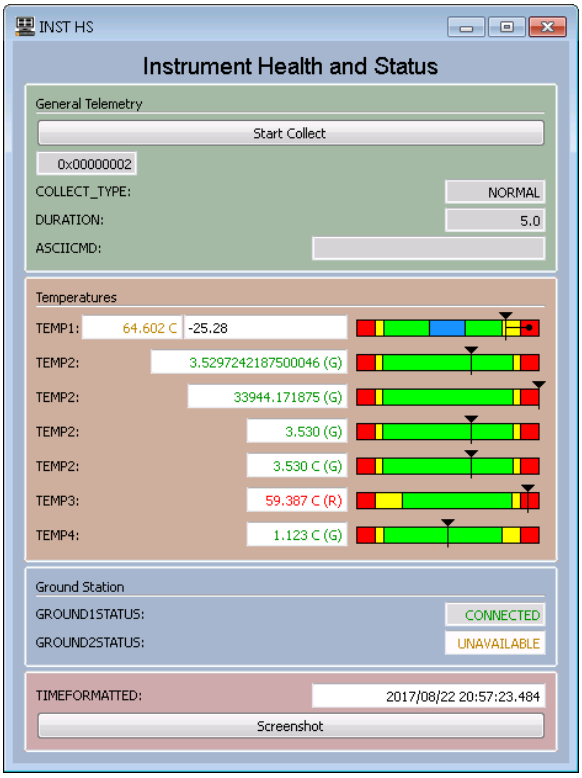

Fig 9. Telemetry Viewer

Packet Viewer provides real-time visualization of every telemetry packet that has been defined. Values within packets are displayed in a simple key-value format that requires no configuration. An autocomplete search bar makes finding values easy.

Telemetry Viewer provides custom telemetry screen functionality with advanced layout and visualization widgets. Tabs, graphs, limits bars, and other animated displays can be quickly created. Also, Telemetry Viewer can autogenerate a base set of screens for every telemetry packet that can be customized as needed.

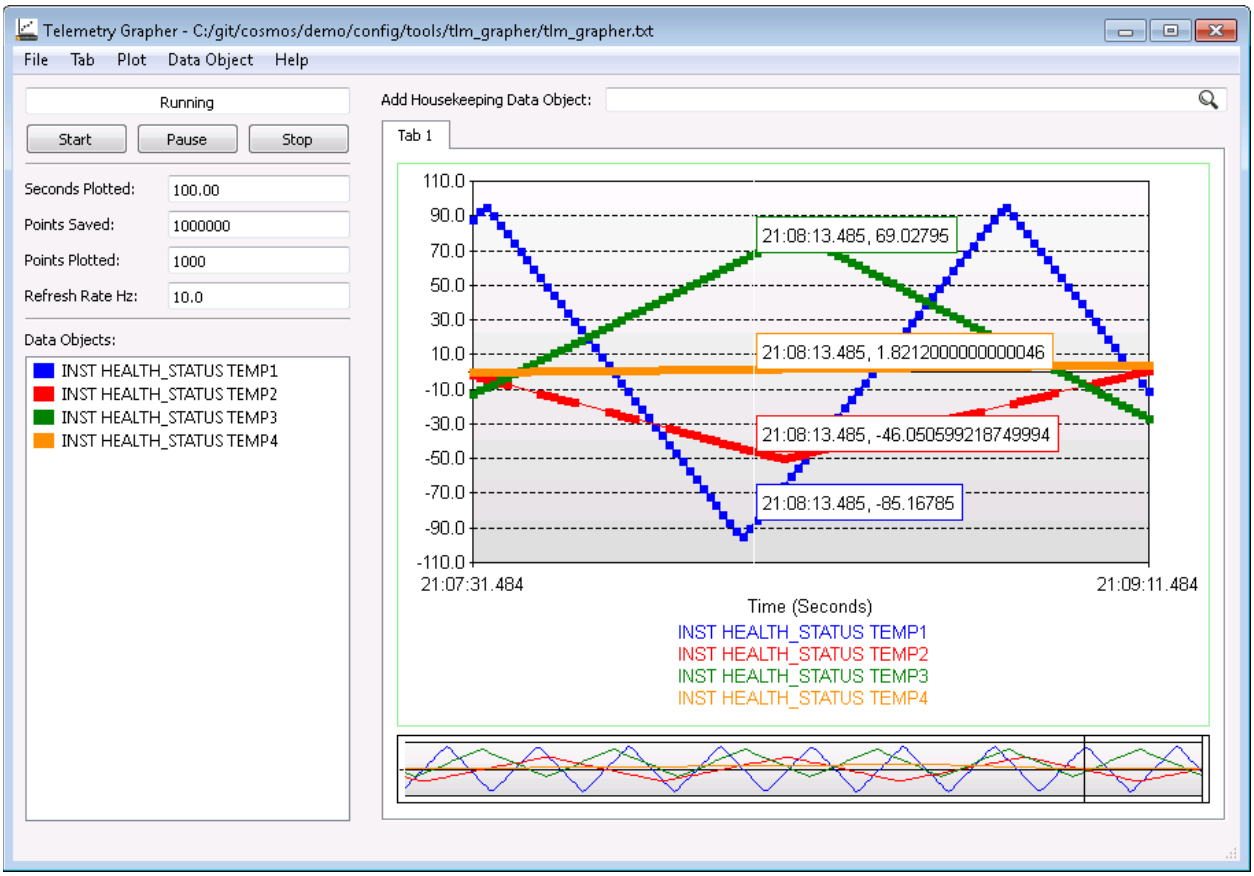

Fig 10. Telemetry Grapher

Telemetry Grapher provides real-time and offline graphing of telemetry data. Supports both line and x-y style plotting, with multiple tabs, plots, and items per plot. Includes built-in analysis functionality to graph min, max, difference, and standard deviation.

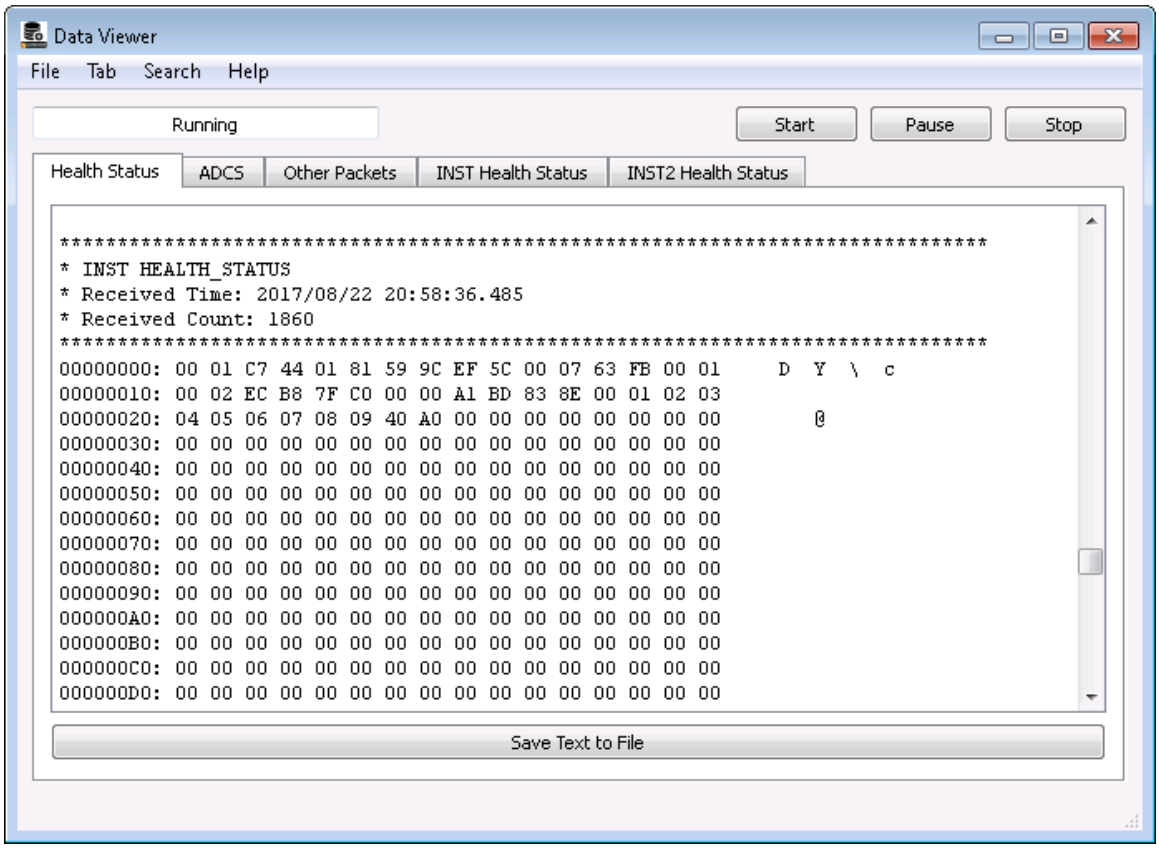

Fig 11. Data Viewer

Data Viewer provides text based telemetry visualization for items that don't fit into other data visualization paradigms. It is great for scrolling log displays and memory dumps.

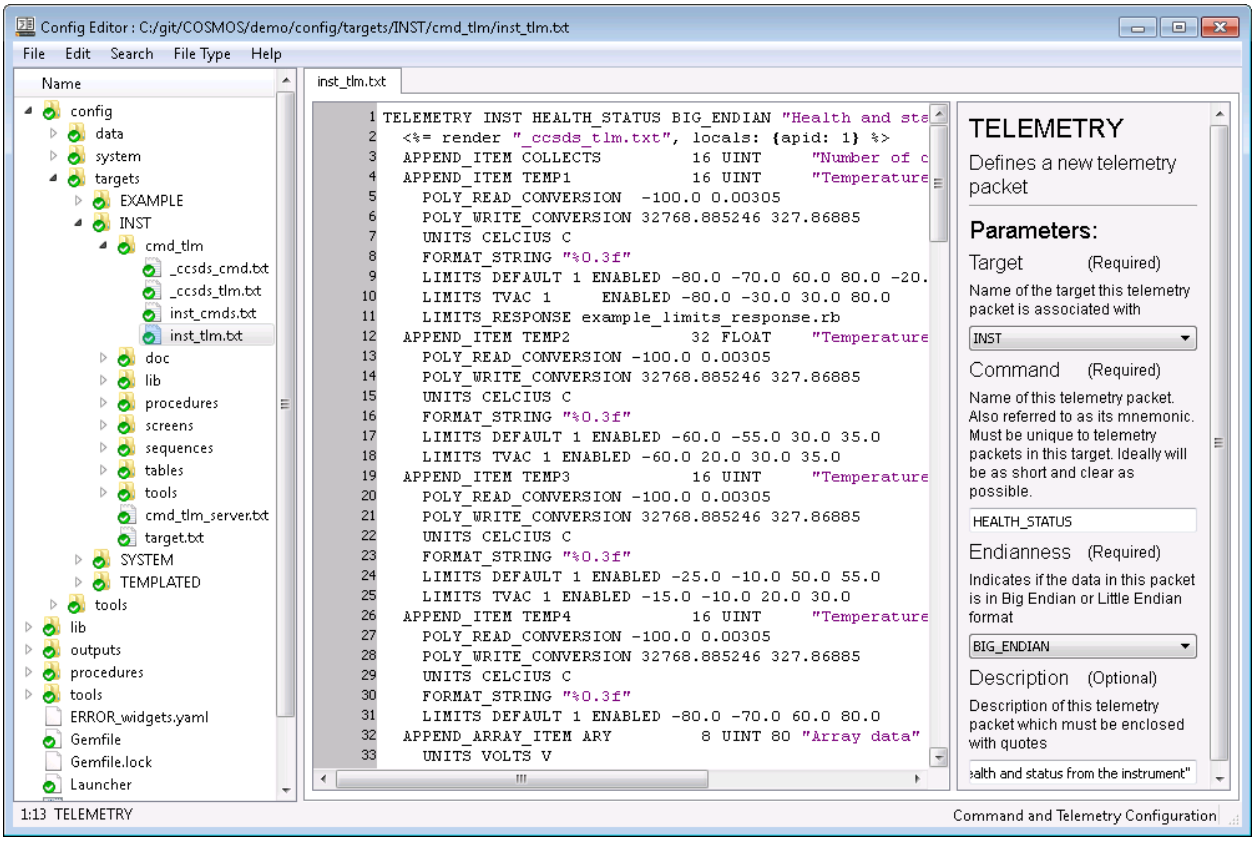

Fig 12. Config Editor

Config Editor provides GUI contextual help when editing COSMOS configuration files. The left pane allows you to easily navigate the COSMOS project in a directory tree. The right pane provides descriptions and drop downs for the current line being edited.

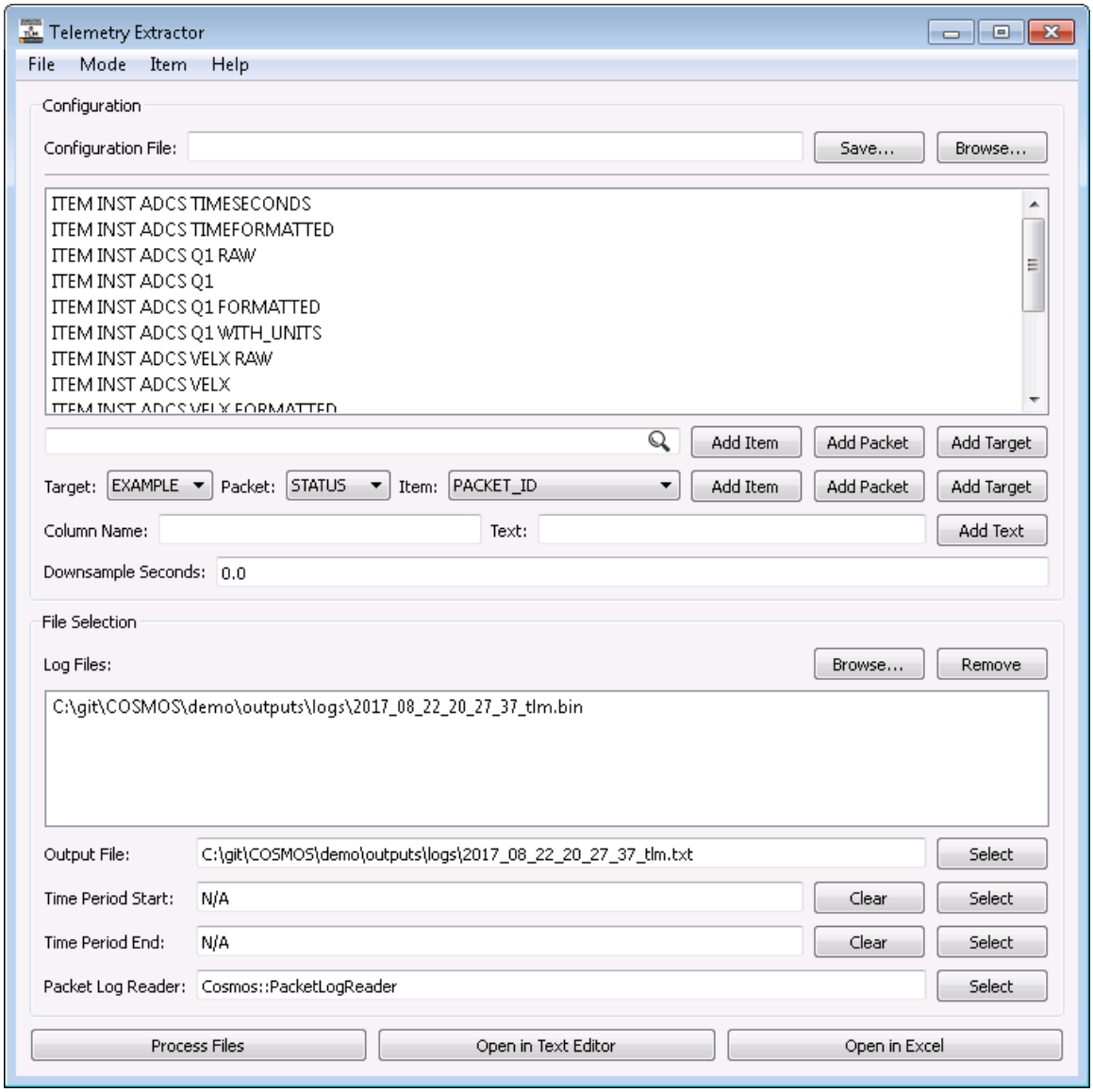

Fig 13. Telemetry Extractor

Telemetry Extractor extracts telemetry packet log files into CSV data. Highly configurable and supports batch processing to output multiple files at once.

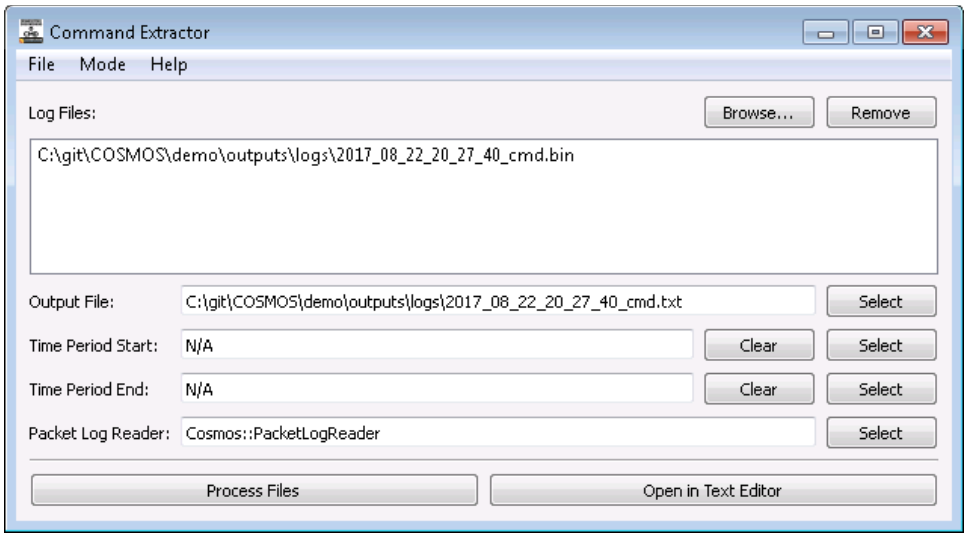

Fig 14. Command Extractor

Command Extractor extracts command packet logs into human readable text.

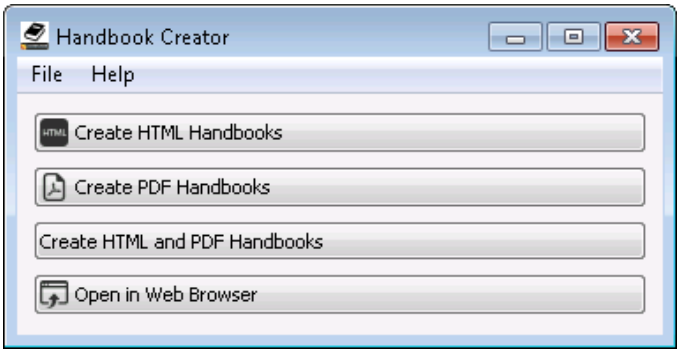

Fig 15. Handbook Creator

Handbook Creator creates html and pdf documentation of available commands and telemetry packets.

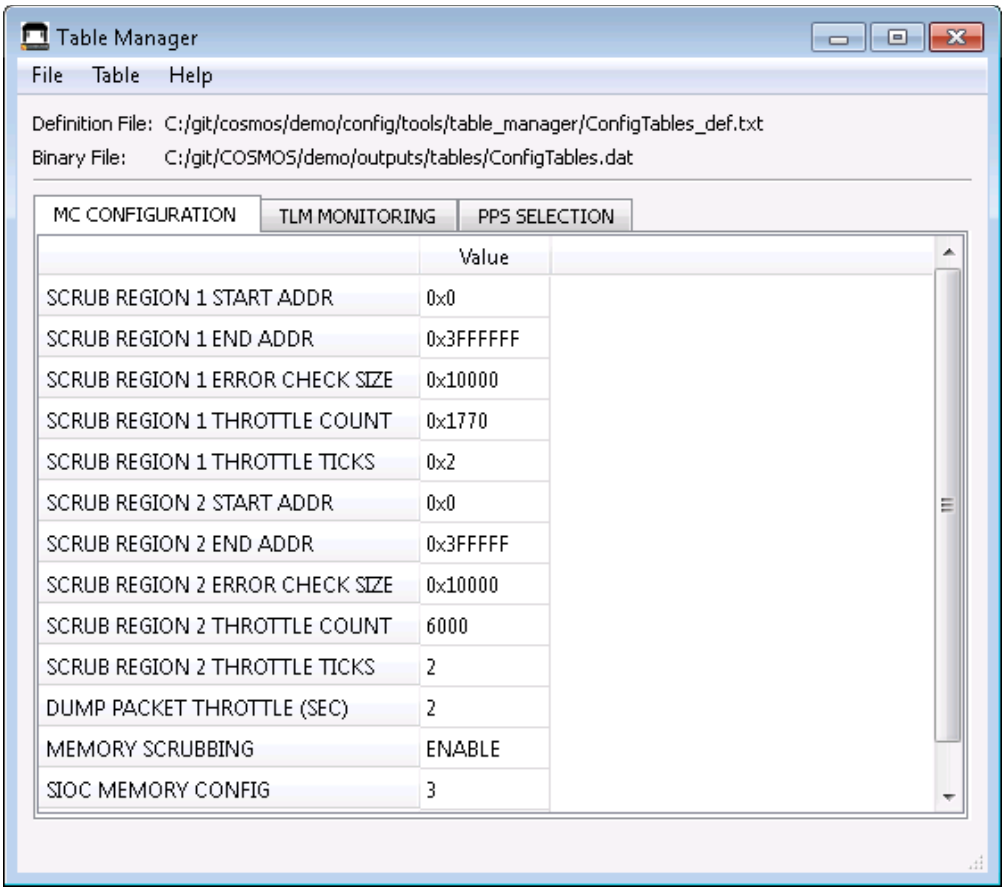

Fig 16. Table Manager

Table Manager is a binary file editor that can be used to create or edit configuration tables or other binary data.

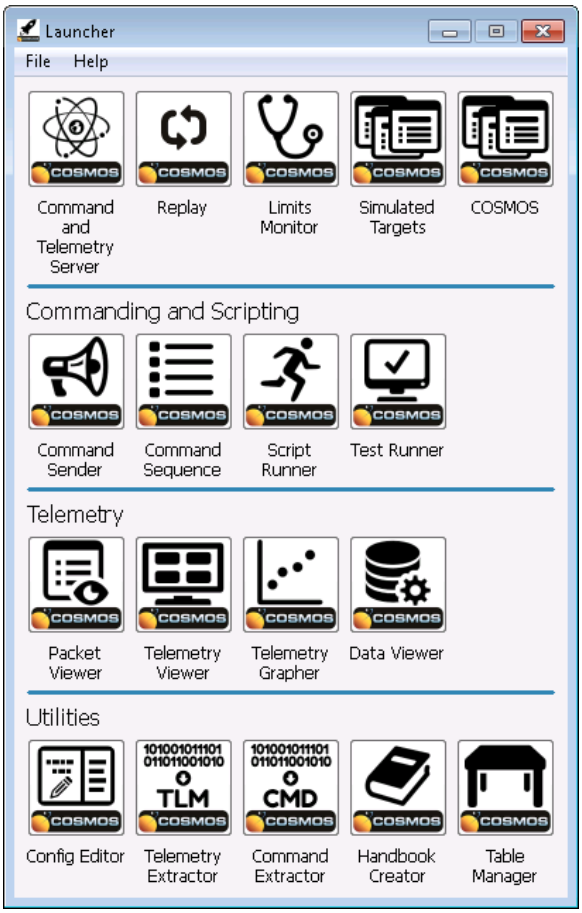

Fig 17. Launcher

Launcher provides a graphical user interface for launching each of the tools that make up the COSMOS system. Supports launching any application that can be started from the command line.

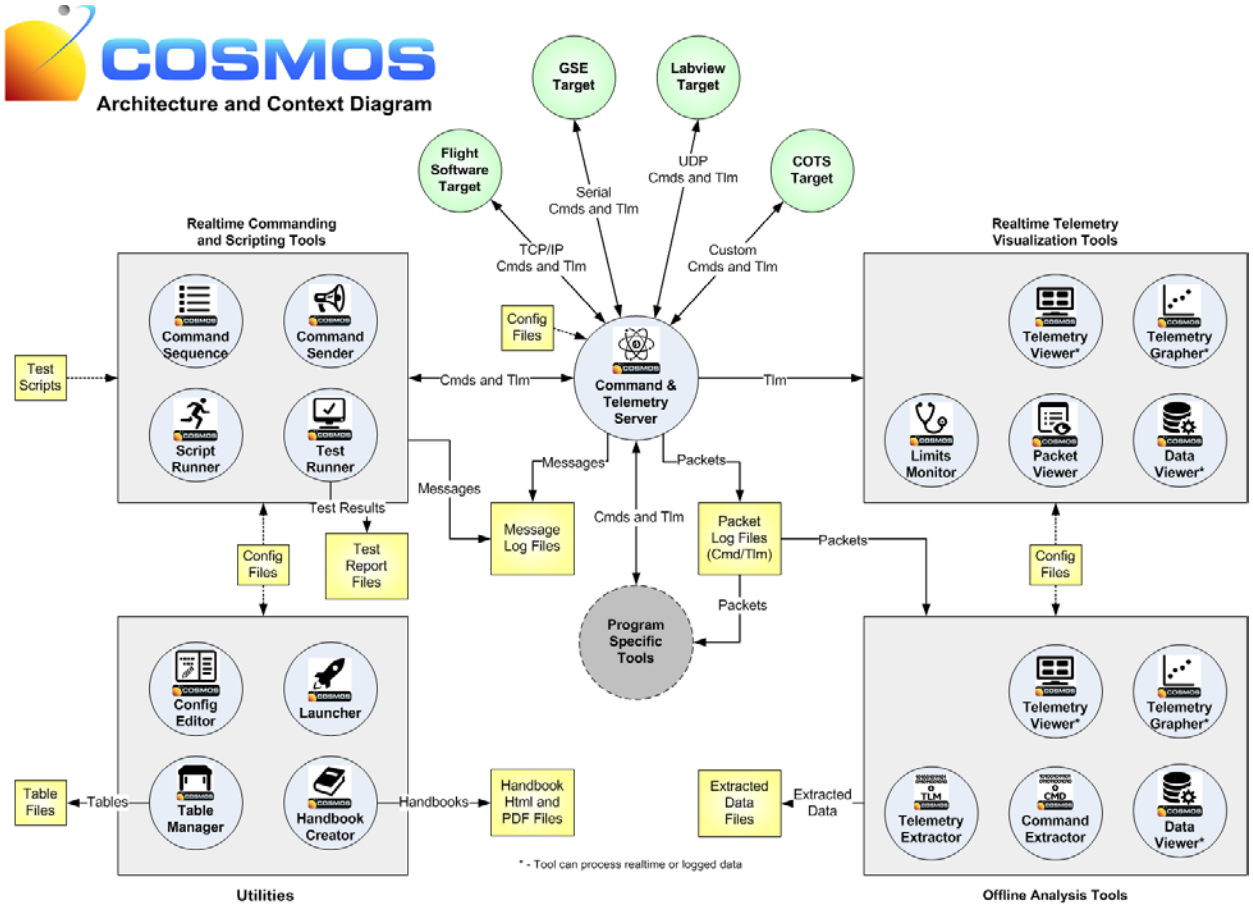

Fig 18. COSMOS Architecture

Fig 18 shows how the 17 applications that make up the COSMOS system relate to each other and to the targets that COSMOS is controlling.

5. CLOUD DEPLOYMENT

Deploying to the cloud provides several benefits for the SSA C2 ground system operator. Once COSMOS has been deployed to the cloud any user can connect to the deployed instance to operate COSMOS. The cloud also provides scalable infrastructure to allow COSMOS to handle increased processing loads by simply scaling the deployed instance. Once on the cloud, COSMOS can interface with existing cloud applications and services much easier than if it were deployed on a standard workstation.

Due to the cross-platform nature of COSMOS, it can be deployed to the cloud in many different instantiations. Amazon Web Services (AWS) provides images for Windows Server, Red Hat Linux and Ubuntu which have all successfully hosted COSMOS. Installation on the Windows Server instance is as simple as running the COSMOS installer and starting the Launcher.

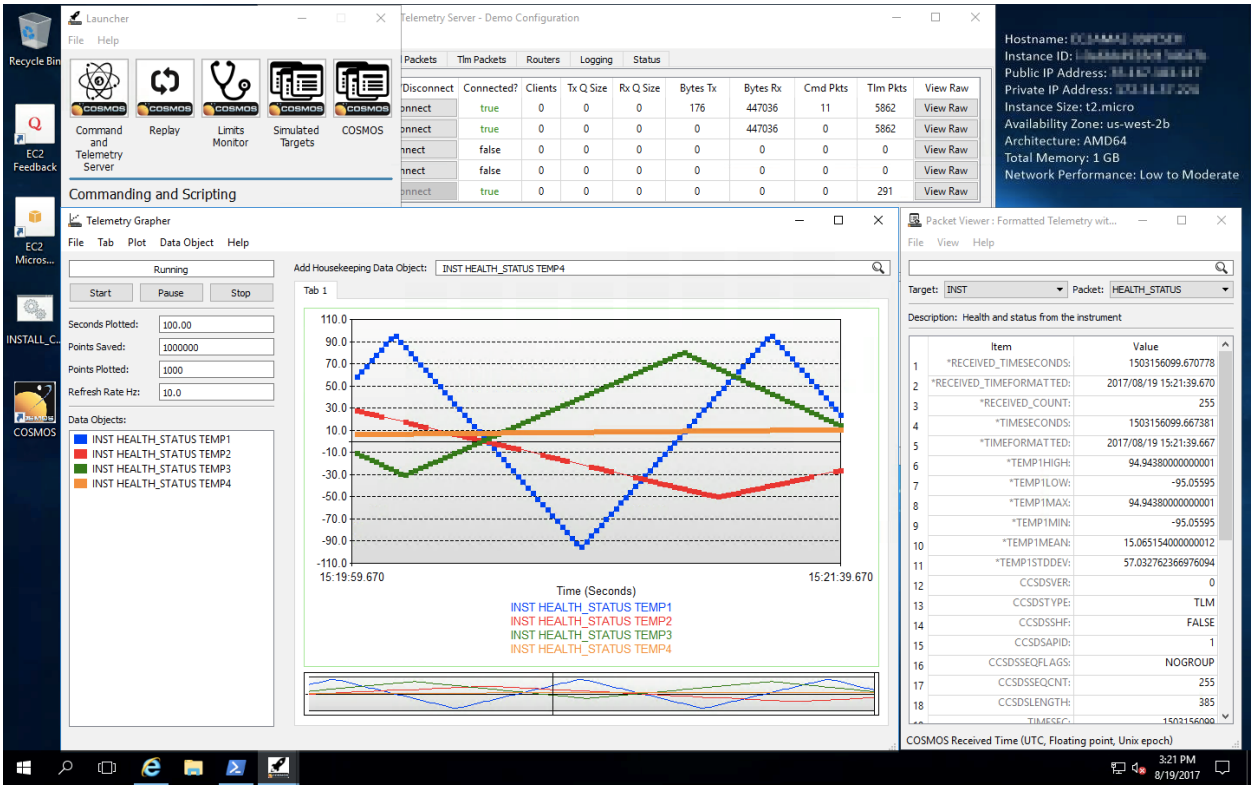

Fig 19. Windows Server AWS

Deploying COSMOS to Red Hat or Ubuntu on AWS requires installing a GUI desktop and then simply running the standard COSMOS Linux installer script. COSMOS has successfully been run through XWindows or VNC on both architectures. Fig 20 depicts COSMOS running on Red Hat Linux with SSH X forwarding to a Mac OS machine running XQuartz.

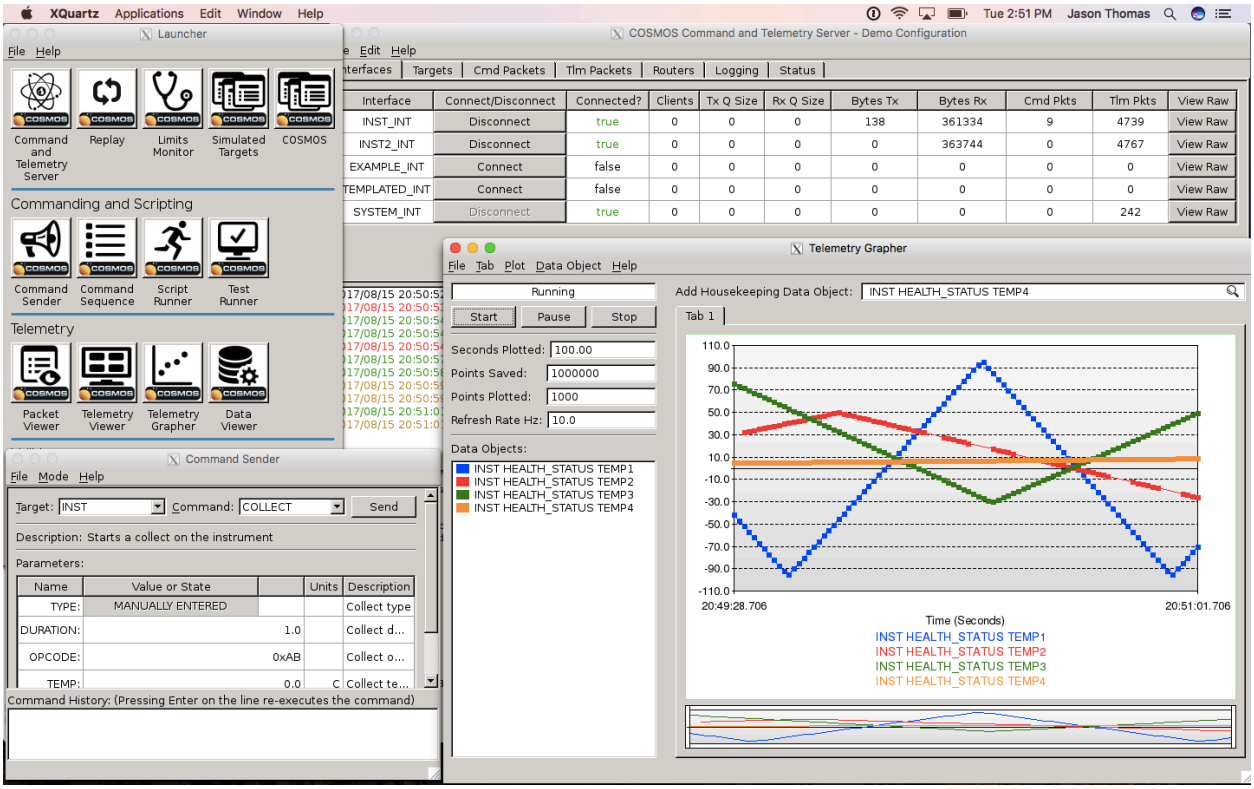

Fig 20. Red Hat AWS X Server

Fig 21 depicts COSMOS running on Ubuntu Linux through a VNC Client. This solution was found to have much better performance than SSH X forwarding.

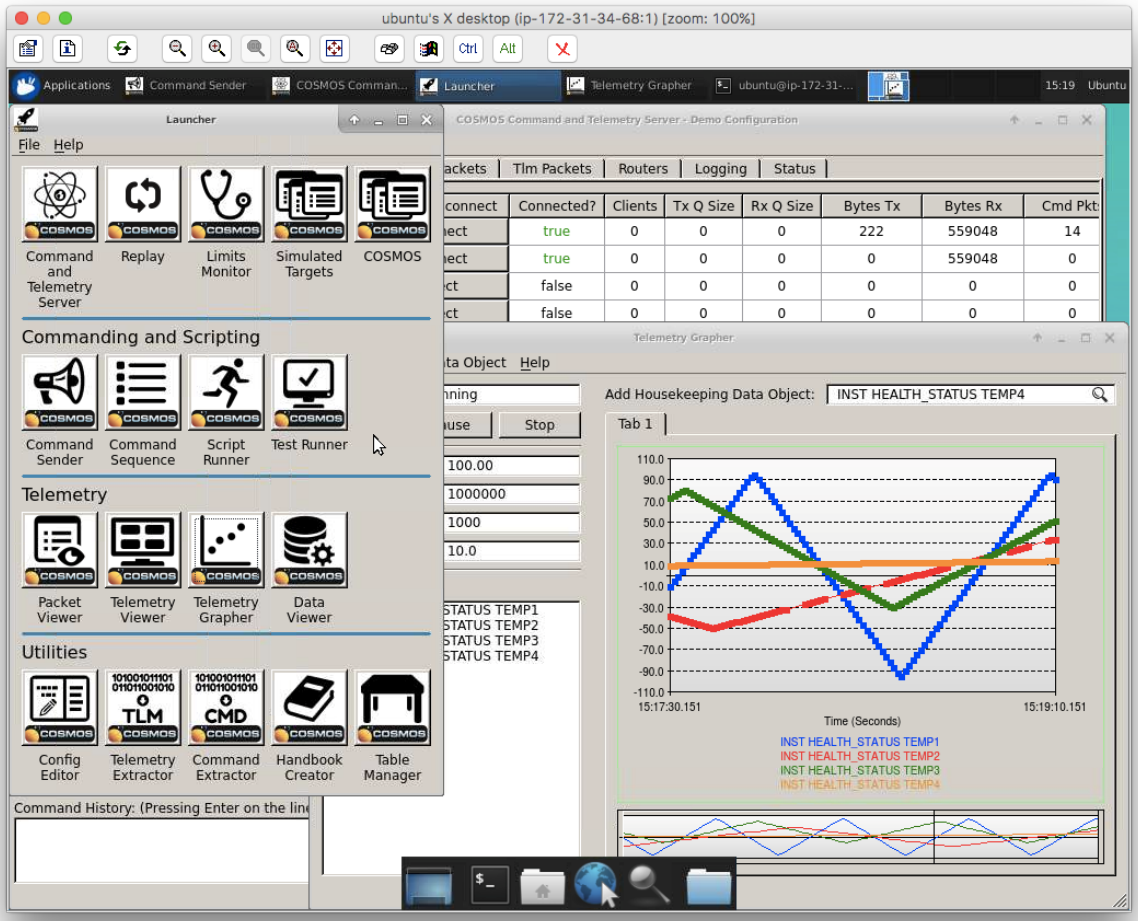

Fig 21. Ubuntu AWS VNC Client

6. CONCLUSION

Ball Aerospace COSMOS is a free and open source command and control system that is immediately available for use. It provides a wealth of functionality, much of which is not even available in expensive proprietary tools. It has been deployed to AWS and is ready for use in operational programs. COSMOS has extensive heritage as it was first developed in 2006 and has since been used to develop and test more than 30 flight programs at Ball Aerospace including GMI, OLI, Kepler, WISE, OMPS, Ares, Orion, and numerous defense programs. Since being open sourced in January 2015 it is now being used with at least 10 major corporations and numerous Universities. For more information and to get started with Ball Aerospace COSMOS please see http://cosmosrb.com.

7. REFERENCES

1. Free Software Foundation, Inc., *GNU GENERAL PUBLIC LICENSE*, Version 3, 29 June 2007