Ground-Based Control of Satellite Clusters

Ross Wainwright

Air Force Research Laboratory
3550 Aberdeen Ave SE
Kirtland AFB, NM 87117

ross.wainwright@vs.afrl.af.mil

Abstract

Air Force Research Laboratory's Distributed Architecture and Simulation Laboratory (DASL) is designed to support the development of new concepts in space systems. One new concept is AFRL's TechSat 21 program. TechSat 21 is a cluster of formation-flying and cooperating satellites. TechSat 21 poses significant challenges to satellite operations. The ground control station must be capable of monitoring and commanding a 'virtual" satellite. In the last few months, satellite simulations, intelligent ground systems, and visualizations have been built and pulled together into an end-to-end system. This system passed an important demonstration milestone. The continuing effort is to create a more precise model of the satellite cluster in the intelligent ground station.

Keywords

Satellite Operations, Expert System, Telemetry, Commanding, Satellite Cluster

Introduction

The Air Force Research Laboratory's Technology Satellite of the 21st Century Program (TechSat 21) envisions a system of formation-flying satellites. A cluster of cooperating satellites enables the accomplishment of new types of missions such as space-based interferometers and sparse aperture radar. The TechSat 21 program office plans to launch a three-satellite cluster for a proof-of-concept flight in 2003. (TechSat 21)

AFRL's Distributed Architecture and Simulation Laboratory (DASL) is designed to support the development of new concepts in space systems. DASL's current focus is the TechSat 21 proof-of-concept flight. Within DASL are environmental, payload, & spacecraft simulations, visualization tools, flight systems, and a satellite operation segment. These systems work together to allow the experimenter to test advanced concepts in a realistic environment.

This paper focuses on satellite operations. Algorithms developed for satellite operations will be transitioned to operational ground control systems to assist Techsat ground operations. The paper begins with a description of current-day satellite operations followed by a discussion of the satellite operations component of DASL.

The first major DASL milestone was a demonstration for the Space Vehicles Director. This demonstration is described.

Traditional Satellite Operations

Satellite Operations are performed to verify and maintain satellite health, to reconfigure and command payloads, to detect, identify, and resolve anomalies, and to accomplish Launch and Early Orbit operations. Telemetry Monitoring, Tracking, & Commanding are the three basic functions of Satellite Operations.

Telemetry is monitored to perform routine satellite health checks, and fault detection, isolation, & resolution (FDIR). Telemetry is used to download the information captured by onboard sensors or other payloads

Tracking determines a satellite's position and velocity. Usually tracking is the process of following the movement of a satellite by keeping the main beam of a ground antenna pointed at the satellite, and measuring the bearing and distance of the satellite with respect to the antenna.

Commanding is the transmission of instructions to a satellite. These instructions (or commands) are used to control or task the operation of a satellite. A few examples of satellite tasking are battery charging, vehicle payload configuration, redundant unit swaps, vehicle maneuver & repositioning, and anomaly recovery.

DASL Satellite Operations

The satellite operations portion of the DASL linked a MatLab simulation of a satellite cluster, an intelligent ground station, and an orbit visualizer.

MatLab Simulation.

ObjectAgent under MatLab was used to simulate a four-satellite cluster. Four were simulated because three are more interesting if one fails.

ObjectAgent builds satellite flight-software simulations using independent software agents. Software agents are excellent candidates on-board satellite autonomous operations. Software agents are modular (allowing replacement and upgrades), collaborative (cooperating satellite clusters), goal-oriented (mission planning), and adaptive (respond to changes). ObjectAgent is a commercial product of Princeton Satellite Systems (PSS). PSS is under contract to the Air Force to enhance ObjectAgent in support of DASL. An essential element of the PSS approach is the agent messaging architecture, which provides a reliable method for agent-to-agent communication both on a single processor and across networks. For the Director's demonstration, each satellite has an orbit propagation agent, attitude agent, and a fuel agent. One satellite also has a Cluster Manager Agent. The Executive Controller has knowledge of all four satellites and handles communication with the outside world.

Each satellite has only eleven mnemonics.

- X, Y, Z positions in the earth centered Cartesian coordinate system (Km)
- X, Y, Z velocities (Km/Sec)
- X, Y, Y thruster forces (N)
- Fuel level (Kg)
- Payload status (Boolean)

Intelligent Ground Station.

The intelligent ground station is Interface and Control System's "Spacecraft Command Language" (SCL). The SCL system integrates expert system technology with procedural programming. SCL accomplished the satellite operation's telemetry monitoring and commanding functions. Since there was no RF link, there was no tracking function.

SCL requires the development of a database, scripts, and rules. The database included the eleven mnemonics sent for each satellite, derived mnemonics, and "state" mnemonics. Altitude is a derived from the x, y, and z positions. State mnemonics are enumerated types. A satellite's state-of-health (SOH) might be "UNKNOWN" (the startup), "NORMAL" (good values), or "ANOMALY" (bad lower state). The states of a satellite's position can be UNKNOWN (the startup), NORMAL (good times and values), ANTICIPATING (good x, y, & z values but with different time tags), GPS_ERROR (basically lost), or LANDED (computed altitude is lower than the earth's surface).

The only script was the initialization script use to set mnemonics to default values.

Rules were constructed for limit checking, to evaluate states, and to compute derived mnemonics. Computation of derived mnemonics also required that the time stamps of each mnemonic used in the calculation be compared. For example, if altitude was to be computed, then it was desirable to have x, y, and z coordinates with the same time stamp and for the same satellite. Rules were also needed to ensure that correct position and velocity vectors were sent to the Visualizer for display. Displayed satellites would behave rather oddly if not all the positions and velocities have the same time stamp.

The receipt of new data causes a rule to fire that executes scripts embedded in the rule. The result may be that one of the lower level states changes. When a lower level state changes, another rule fires that re-evaluates higher level states. The highest states are usually NORMAL or ANOMALY. Lower states are more descriptive. Any other than NORMAL lower level state percolates up to an ANOMALY upper state. A NORMAL highest state indicates that all lower states are NORMAL.

Since SCL has a command line interface, ICS provides connections to third-party graphical user interfaces. In our case, we used LabView to develop a GUI for telemetry display and satellite commanding. The sections of the LabView GUI for satellites #1 and #2 are shown in Figure 1. Satellite #2's fuel level at lower red. This makes satellite #2's fuel state "RED_LINE", and satellite #2's state becomes "ANOMALY".

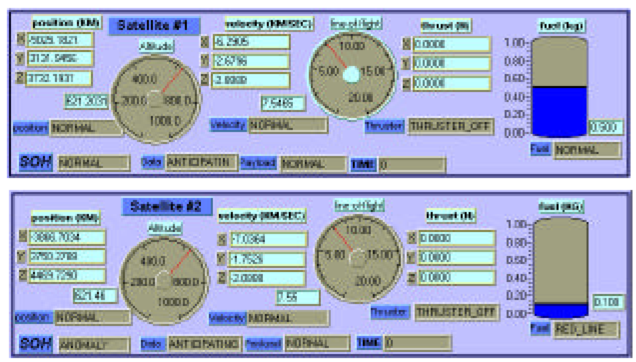

Figure 1 Satellite#1 and Satellite#2 Ground Display

Because the available mnemonics were limited, few conclusions were possible concerning cluster SOH. (1) The four satellites should be in the same plane. (2) A minimum separation between satellite should be maintained. (3) Cluster SOH depended in individual satellite SOHs.

Distances were computed using the Cartesian distance formula. Four points are in the same plane if a determinate of a matrix made up of their positions is zero.

$$\begin{vmatrix} \mathbf{X_1} & \mathbf{Y_1} & \mathbf{Z_1} & \mathbf{1.0} \\ \mathbf{X_2} & \mathbf{Y_2} & \mathbf{Z_2} & \mathbf{1.0} \\ \mathbf{X_3} & \mathbf{Y_3} & \mathbf{Z_3} & \mathbf{1.0} \\ \mathbf{X_4} & \mathbf{Y_4} & \mathbf{Z_4} & \mathbf{1.0} \end{vmatrix} = \mathbf{0.0}$$

The cluster section of the LabView GUI is shown in Figure 2. Cluster state is "ANOMALY" because the SOHs of satellite #1 and satellite #2 are "ANOMALY". Ellipse state is "ANTICIPATING" because the time stamps of the four satellite are not all the same.

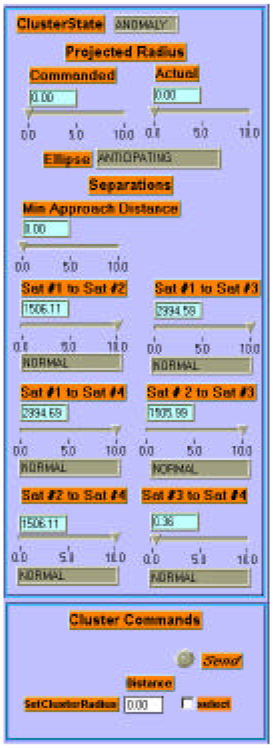

Figure 2. Cluster

The Intelligent Ground Station allowed the user to select and send commands back to individual satellites in the MatLab Simulation. Three commands would cause thrusters to fire. Two others will enable or fail a satellite. These five commands were collected into one GUI for each satellite (Figure 3).

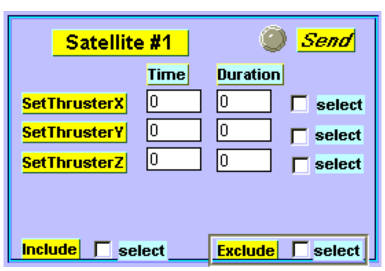

Figure 3. Satellite 1 Commanding

Orbit Visualizer.

Visualizer uses PostGres database to store cluster position, velocity, and thrust. SCL needed to connect to PostGres and populate tables to allow display of the four satellite cluster.

The Demonstration.

The Director's demonstration showed that all the pieces work together. ObjectAgent send telemetry from each satellite to the Intelligent Ground Station where the telemetry was analyzed and displayed. The ground station in turn gathered position and velocity vectors and sent them to the Visualizer program where four satellites were displayed orbiting around the earth (Figure 4).

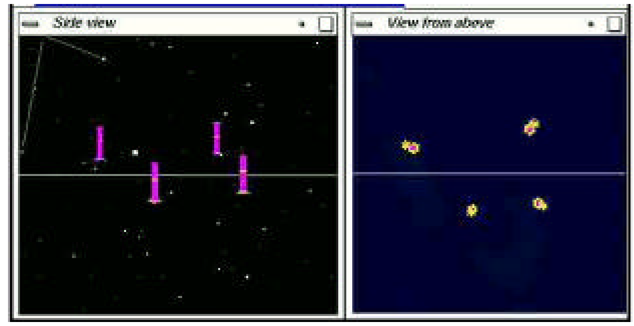

Figure 4. Visualizer Before Failure

The ObjectAgent simulation included GUI that allowed the operator to cause a component to fail. For the demonstration, satellite #1 payload was "failed" (Figure 5).

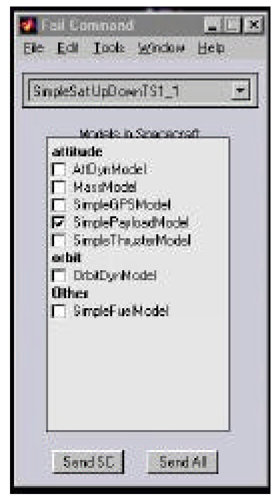

Figure 5. Fail satellite #1 Payload Model

The failure of the payload resulted in satellite#1 SOH changing to "ANOMALY" (Figure 6). SOH is "ANOMALY" because payload state is "FAILED". Thruster state is "THRUSTER_ON" because x thrust is greater than 0. Fuel level has diminished but not below yellow or red limits.

Figure 6. Satellite #1 Payload Failure

The Cluster manager responded to satellite #1's payoad failure by the repositioning the four satellites. The Visualizer shows satellite #1 moving away from the cluster and the remaining three moving into an equilateral triangle (Figure 6).

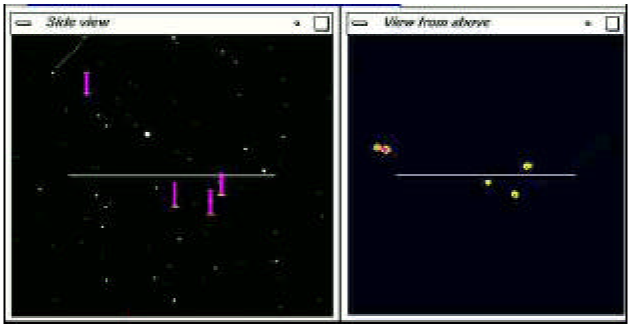

Figure 6. Visualizer after Failure

After the Demonstration

A successful demonstration is completed. The goal now is real-life and real-time performance.

Realistic flight system

ObjectAgent is being ported to Enea Systems/OSE real-time operating system. The baseline satellite cluster will be collection of single-board-computers (PowerCore-6750) mounted in two VME chassis and running ObjectAgent flight software. Each VME chassis is split into four 5-slot back planes. One 6750 will be installed into each 5-slot back plane. Thus, the 6750s will be electrically connected, but logically separate. Each board represents one satellite flight processor. Ethernet connections through a switch represent cross-link communications.

Realistic telemetry link.

One of the 6750s will have a SBS-4416 telementry encoder board installed with it in its 5-slot bus. Thus, frame-formatted/pulse code modulated telemetry can be piped to a commercial-off-the-shelf telemetry analysis system and then to the SCL workstation for display and analysis.

Cluster Model.

For the demonstration, SCL reacted to telemetry produced by ObjectAgent. SCL requires a more detailed model of the cluster and of each satellite. This model should be developed in concert with the development of the flight system. [It will be refreshing to work with the flight software engineers. Usually, the ground-based expert system is programmed after launch using out-of-date documentation and guesswork. (Zetocha 1997).]

The model will be complex. Hundreds (possible thousands) of mnemonics are needed to monitor the state-of-health and performance of each satellite. A representative spacecraft's flight software consists of seven bus subsystems and the payload. The bus subsystems are Guidance, Navigation & Control (GNC); Communications (COMM); Command & Data Handling (C&DH); Electrical Power (EPS); Thermal (THERM); Structures & Mechanisms (STRUCT); and Propulsion (PROP). Bus subsystems support the payload in several ways such as pointing, controlling temperature, supplying electrical power, and commanding. (Larson 1993)

The products of a previous domain analysis provided structure to satellite flight software. In this previous project, a mechanism was required to aid the application engineer in tailoring and specifying the generic architecture to meet the requirements of a specific project. The mechanism was the Decision Tree shown at its highest level below in figure 7. (Wainwright 1998). Potentially, the Domain Tree could be adapted to provide a hierarchy of satellite states.

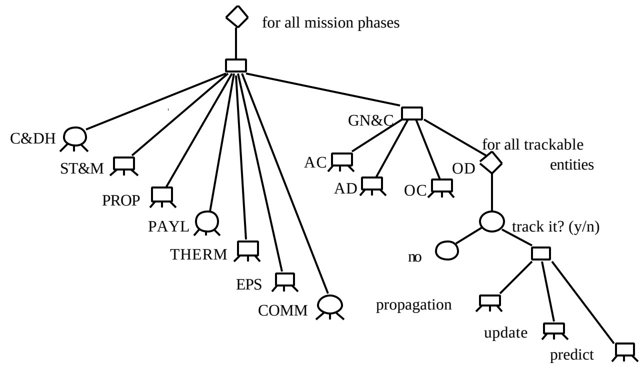

Figure 7 Decision Model

Using the model, SCL will watch and make observations on the performance of the on the individual satellite level and at the cluster level. More importantly, SCL will suggest commands at the cluster and the individual satellite level. Commands will be transmitted via as realistic as possible link to the Cluster Manager. The result of commanding will be predicted and the predicted result will be compared the actual result using performance validation metrics.

Conclusions

TechSat 21 shifts the satellite operations paradigm. Just communicating from the ground is a problem. All of satellites in the cluster are within the beam width of one ground station.

The DASL environment allows us to devise and test new concepts in Satellite Operations. In the last few months, satellite simulations, intelligent ground systems, and visualizations have been built and pulled together into an end-to-end system. This system passed an important demonstration milestone. The continuing effort is to create a more precise model of the satellite cluster in the intelligent ground station.

Acknowledgements

Derek Surka of Princeton Satellite Systems and Lance Self of AFRL developed the ObjectAgent simulation. Mark Tolefson and Jeremy Rosenstock of Nichols Research

developed the Visualizer and the PostGres database, respectively. Paul Zetocha lead the Satellite Operations demonstration.

References

TechSat 21 Space Missions Using Satellite Clusters http://www.vs.afrl.af.mil/vsd/techsat21/

P. Zetocha, R. Wainwright, "Case Study – Expert System for Satellite Control", Proceedings of the 9th Annual Software Technology Support Conference, Salt Lake City UT, Apr 1997.

Larson, W.J. and Wertz, J,R. (eds) Space Mission Analysis and Design, Second Edition, Microcosm Inc, Torrance, CA, and Kluwer Academic Publishers, London, The Netherlands, 1993

R. Wainwright, "An Application Engineering Workbench for Tailoring Ada Flight Components", SIGAda'98, Washington D.C., October 1998