STATE MODELING AND PASS AUTOMATION IN SPACECRAFT CONTROL

Jim Klein, Dan Kulp, Bob Rashkin

ABSTRACT

The Integrated Monitoring and Control COTS System (IMACCS) was developed as a proof-of-concept to show that commercial off-the-shelf (COTS) products could be integrated to provide spacecraft ground support faster and cheaper than current practices. A key component of IMACCS is the Altair Mission Control System (AMCS), one of several commercial packages available for satellite command and control. It is distinguished from otherwise similar tools by its implementation of Finite State Modeling as part of its expert system capability. Using the Finite State Modeling and State Transition capabilities of the ALTAIR Mission Control System (AMCS), IMACCS was enhanced to provide automated monitoring, routine pass support, anomaly resolution, and emergency "lights on again" response. Orbit determination and production of typical flight dynamics products, such as acquisition times and vectors, have also been automated.

KEY WORDS

Automation, IMACCS, state model, state transition

INTRODUCTION

IMACCS grew out of the Goddard Space Flight Center (GSFC) Mission Operations and Data Systems Directorate (MO&DSD) Reusable Network Architecture for Interoperable Space Science, Analysis, Navigation, and Control Environments (RENAISSANCE) effort. The purpose of RENAISSANCE is to re-engineer the process by which the MO&DSD builds and operates ground data systems, so that these processes become faster, cheaper, and more flexible than has been the case. The use of COTS products was one area selected for investigation. To demonstrate the feasibility of this approach, the IMACCS team undertook and accomplished the task of building a working COTS-based ground support system in 90 days. The original system mirrors the existing ground support system for the Solar, Anomalous, and Magnetic Particle Explorer (SAMPEX) spacecraft. During the IMACCS implementation, the potential for significant automation of ground support became apparent, and the addition of extensive automation was a major component of the second phase of development. This paper describes the main tool used in that process,

finite state modeling, as well as the results of the automation process itself. A detailed description of IMACCS can be found in Scheidker, et. al., 1996.

FINITE STATE MODELING

State Modeling is a process in which a system is represented by a hierarchy of "state vectors" to give a meaningful report of the system's condition at any time. At the lowest level, the state is usually defined by the values of several individual items of telemetry, system variables, and/or parameters derived from telemetry items and system variables. At higher levels, the state is defined according to the states of lower level systems, often in combination with other telemetry items and derived parameters. In this way, the model is built up to a meaningful one-line description of the state of the entire system (in this case, SAMPEX).

The levels of the state hierarchy must be meaningful to the user. Systems called out to the same level should have the same degree of interest to the person monitoring their status. The states designated for a system should likewise be conditions of interest to the user. So, for example, if a coil can have current flowing in either direction, that can define one state or two, depending on whether or not the direction of current flow is considered significant.

Consider, for example, the SAMPEX attitude control system (ACS) functional model. The ACS state is determined by the states of its 2 main subsystem - the sensor and actuator subsystems. These, in turn, are determined by the states of the individual sensors and actuators, which are determined by the values of appropriate telemetry items. The Altair Design Reference Language (ADRL) defining the ACS functional model is illustrated in the Appendix.

While the functional state provides information on the health of a spacecraft subsystem, it says nothing about the actual performance of that subsystem. We can know that wheel speeds, torquer currents, battery voltages, etc., are all within nominal limits and yet have little information about the actual operation of the spacecraft. This knowledge gap is filled by the use of dynamic state modeling, whereby parameters relevant to the flight dynamics of the vehicle describe such items as position, velocity, attitude, attitude motion, and control modes. For IMACCS, the ACS dynamic model describes the system in terms of lighting, magnetic field, and control mode.

STATE TRANSITION

The use of state modeling for data monitoring facilitates the use of another powerful process, State Transition Modeling, to automate spacecraft control. This technique models commands as transition vectors between two vehicle states. These can be initiated

manually, time-based (typically for routine pass activities), event-based (for reacting to the detection of a specified state, expected or not, by initiating a transition to a different state). Transitions are defined in an object-oriented database and typically comprise verification of initial (entry) state, constraint checking, the transition vector, and verification of success. A transition failure vector, i.e., action to be taken should the target state not be achieved, may also be defined, as may a transition success vector and an entry failure vector. The latter initiates a course of action if the system is not in the proper entry state for the desired transition. The transition vector itself may be a string of commands or a procedure written in the ALTAIR Procedural Automation Language for Spacecraft (PALS).

ADVANTAGES

State modeling uses the computer to do what it does best, liberating humans to do what they do best. Spacecraft data monitoring has typically been effected by one or more people observing several displays. Before launch, engineers must decide upon a very small subset to be monitored from among the thousands of individual telemetry items available. This is modified as operational experience provides more insight to the vehicle's inflight performance, as hardware degrades, or as the mission's priorities are changed. Moreover, the importance of any one item's value may be dependent upon several other parameters. The operations engineer must either know what combinations of readings are meaningful or change limits based upon known or anticipated operational modes. In effect the engineer carries a state model in her head (or on paper), and evaluates the state of the spacecraft according to this model. Since this is done by visual inspection of displays, it can perhaps be done once or twice per minute. This is a singularly inefficient use of the human brain. It is also risky, since potentially important short-term events might be missed. (Conversely, however, it should be noted that people are better able to recognize an unimportant transient event for what it is, probably the major weakness of the state recognition process. This can be overcome by combining the state recognition process with a rule-based expert system. See below.)

The use of state modeling eliminates these problems. The state recognition process monitors every telemetry item and determines the vehicle state for every telemetry frame. Thus, nothing is missed, the danger of human error is minimized, and the engineer's time is freed for more useful activities.

Together with the state transition process (see above), the state recognition process also provides significant opportunity for automation. Moreover, the similarity of the process to the normal human mode of incorporating experience facilitates the evolution of the monitoring and control system. Operational experience always results in the modification of monitoring and control activities. In the context of state modeling, states of interest not

described pre-launch are identified and included in the model. In addition, operations engineers continually identify ways to improve procedures and automate processes. State modeling and transition are intuitive processes that facilitate rapid changes, requiring training in neither programming nor expert systems. These concepts are revisited and illustrated with examples below.

CONCERNS

The main concern in using finite state modeling for monitoring and control arises from its single-frame nature. A single erroneous telemetry value can cause the system momentarily to indicate an undesirable state, and, for example, one would normally not want to place the spacecraft in safehold based on a single reading. This problem is addressed in part by data quality checks, which help increase confidence in actions taken in response to anomalous data. In addition, the anomaly can be required to persist through several telemetry frames before action is taken. The state transition process operates in conjunction with a rule-based expert system, or "inference engine," which executes the actual transitions. The inference engine provides the capability to require persistence of an undesirable state before action is taken. It also provides more sophisticated trending capabilities, giving the engineer a great deal of flexibility in defining responses. This area is currently being addressed in depth by the IMACCS team.

AUTOMATION

Operations automation can be divided into 5 classes: data monitoring, routine pass activities, anticipated contingency response, emergency response, and product generation. The IMACCS prototype demonstrates the feasibility automation in all of these areas.

1. DATA MONITORING

The first application of state modeling in IMACCS was the automation of routine data monitoring, as described in Section 3. The IMACCS team, consulting with subsystem engineers and Flight Operations Team (FOT) members, modeled the SAMPEX vehicle according to its major subsystems -- Power, Thermal, Communications and Data Handling (C&DH), Experiment, ACS, and Mechanical. The state of each of these, in turn, depends on the states of its own subsystems, and so on. Note that state modeling permits all manner of system decomposition. For example, we have modeled SAMPEX as the sum of Attitude Control, Power, Instruments, Flight Software, etc. An alternate decomposition might be Science Operations, Bus Operations, Communications, etc. The key is that the decomposition needs to make sense to the user (operator, scientist, engineer). Several alternatives can be defined simultaneously. When the model is complete, the operations engineer no longer has to monitor, for example, the speed of the reaction wheel. Instead,

she simply observes the state of the actuator subsystem. Even this, however, is unnecessary, as long as the state of the ACS is known and desirable. This is leading, of course, to the observation that only one parameter, the state of SAMPEX, must be watched. It is only when this is unknown or undesirable that any other states must be investigated. In fact, as shall be seen, there is ultimately no need for a human to monitor even this one item.

2. ROUTINE PASS AUTOMATION

Experience has shown that, after launch and early mission checkout, 95% of operations are completely routine. For SAMPEX, besides data monitoring, a typical pass involves an initial communications check, accomplished by sending a "no-op" command, dumping an onboard table, and loading an updated table. All these can be scheduled as time-based state transitions. To load the table, for example, the state transition process first verifies the correct entry state, "TABLE_INACTIVE," and then executes the transition to the target state, "LOADING," identified by values of the onboard command counter, a "TABLE_ACTIVE" value in the telemetry, and several other telemetered status indicators. The actual transition vector is a PALS procedure consisting of three spacecraft commands needed to load the table. Upon verification of success, the success vector is executed. This is a PALS "MonitorStateEntry" command instructing the state transition process to perform the transition "TABLE_COMMIT" (write from the buffer to the table's actual memory location) upon verifying attainment of the state "TABLE_LOADED." The table dump transition is similar.

Failure vectors can be executed if the subsystem is not in the expected entry state at the scheduled time, or if the transition fails to achieve the target state. For IMACCS, failure vectors are simple console messages and log recordings. They could, however, execute some corrective action or notify appropriate personnel of the failure.

3. ANTICIPATED CONTINGENCY RESPONSE

Approximately 3% of spacecraft operations involve responses to anomalies that are well-known and documented. During a mission, particularly in the early stages, unanticipated situations invariably arise. Typically, spacecraft engineers analyze the problem and generate a response to be carried out by operations personnel in the event of a recurrence. This response, usually execution of a spacecraft command or series of commands, then becomes part of normal operations. The IMACCS team modeled an automated response to one such SAMPEX contingency. Occasionally, telemetry indicates that the Heavy Ion Large Telescope (HILT) isobutane flow regulator valve is open. Although this is an erroneous telemetry item, the proper response was determined to be a command to close the flow register. This correctly resets the telemetry and, presumably, would close the

valve should it actually be open. Operations engineers now monitor the flow register as part of routine operations, and execute a command script to correct the anomaly when detected.

The state modeling and transition processes are well suited to automate such a response. It is a simple matter to configure the state transition process to monitor the HILT state and transition to the state "NOMINAL" upon detecting the state "FLOWREG_OPEN." The transition vector is a single spacecraft command. While this example is quite simple, the principle easily applies to any situation where there exists a documented response to known anomalous state.

4. EMERGENCY RESPONSE

With proper application of state modeling techniques as described above, the ground support software can support the mission without human intervention, through routine pass activities and known contingencies. This is not enough, however, to allow complete "lights out" operation. The system must also be able to recognize a condition for which it is not prepared, and take appropriate action, which will always include calling for human intervention. In IMACCS, this is demonstrated by simulating an ACS anomaly -- specifically a Digital Sun Sensor (DSS) failure. The state transition process, monitoring the DSS for the state "X_DEAD," executes the transition "REACT_DIGITAL_SUN_SENSOR_FAILURE." The transition vector is a PALS procedure that in turn executes several other scripts. The first is an EXPECT* script that dials a pager and sends a coded message. Next, a UNIX script invokes the AMCS archiving and playback functions to retrieve the latest 24 hours of data relevant to the failed system and enter these into a trending tool (BBN Probe) in order to have plots waiting when the engineer arrives. The capability exists, of course, for far more sophisticated responses.

5. PRODUCT GENERATION

Automated product generation is covered in detail in another paper (Lin, [1996]). It is include here for completeness. Currently, flight dynamics product generation activities involve significant human interaction. flight dynamics products include ephemerides, station acquisition data, calibrations, orbital predictions, etc. In IMACCS, orbit determination and orbit-related product generation are accomplished in large part using Satellite Tool Kit (STK®) from Analytical Graphics Corporation. STK, however, is very highly interactive, and, for routine production, it is desirable to minimize the need for user interaction. The IMACCS team identified XRunner®, from Mercury Interactive

* EXPECT is a public domain program, implemented in the Tool Command Language (TCL). See, for example, Welch, [1995].

Corporation, as a tool to accomplish this. Using a simple PERL script as an executive, and taking advantage of the UNIX "cron" function, which will initiate a process at a preset time, and running STK under XRunner, the team has fully automated routine orbit product generation.

CONCLUSION

Significant automation of operational spacecraft support is possible and Finite State Modeling is an excellent tool for its accomplishment. IMACCS is a relatively simple prototype, but it clearly demonstrates all the capabilities needed for true "lights out" operation. In actual operation, state modeling will be more extensive and more state transitions will be defined. Training of the expert system begins at the bench test level, and continues throughout the premission period, so that, at launch, it is prepared to handle almost all of routine operations. As the mission progresses, the system will be taught (by operations personnel, who are the experts) to assume increasingly broad responsibilities for spacecraft operations until it can be left on its own with confidence.

ACKNOWLEDGMENTS

The work described in this paper was accomplished at the Goddard Space Flight Center (GSFC) under contract NAS 5-31500. The IMACCS effort was carried out primarily under the inspired leadership of Mike Bracken, formerly of NASA/Goddard Space Flight Center (GSFC) and Gary Meyers of GSFC, leader of the RENAISSANCE effort. Sue Hoge of GSFC directed much of the automation work. Bob Connerton of GSFC also provided direction and did much to publicize the effort.

APPENDIX

STATE MODEL EXAMPLE

Figure A-1 shows portions of the ACS functional state model. Some states and items are incomplete due to space considerations. They are analogous to other states and items. Complete models can be obtained from the authors.

Figure A.1 ACS Functional State Model

```
SYSTEM: ACS_HEALTH
  SYSTEM: SENSOR
    SYSTEM: DIGITAL_SUN_SENSOR:
      STATE:  NOMINAL_SUN
          ITEM:   ACECFLG_RAW
                VALUE:          0
                TOLERANCE:    0
                DESCRIPTOR:   ECLIPSE_FLAG_OFF
          ITEM:   ACSUNPFL_RAW
                VALUE:          1
                TOLERANCE:    0
                DESCRIPTOR:   SUN_PRESENCE _ON
          ITEM:   ADSSXER
                VALUE:          129
                TOLERANCE:    14
                DESCRIPTOR:   X_AXIS_ERROR
          ITEM:   ADSSZER
      END_STATE
      STATE:  NOMINAL_SHADOW
      END_STATE
      STATE:  X_DEAD
           ITEM:   ACECFLG_RAW
                VALUE:          0
                TOLERANCE:    0
          ITEM:   ACSUNPFL_RAW
                VALUE:          1
                TOLERANCE:    0
          ITEM:   ADSSXER
                VALUE:          0
                TOLERANCE:    14
      END_STATE
    END_SYSTEM
    SYSTEM:  MAGNETOMETER
       STATE:  NOMINAL
          ITEM:  AMAGX
                VALUE:          0.0
                TOLERANCE:    9.8
                DESCRIPTOR:   X_AXIS_SIGNAL
          ITEM:   AMAGY
          ITEM:   AMAGZ
      END_STATE
    END_SYSTEM
   /* SENSOR STATES */
    STATE:  NOMINAL_SUN
       ITEM:   DIGITAL_SUN_SENSOR_STATE
             VALUE:  NOMINAL_SUN
       ITEM:   MAGNETOMETER_STATE
             VALUE:  NOMINAL
    END_STATE
    STATE:  NOMINAL_SHADOW
    END_STATE
/*   END SENSORS, START ACTUATORS */
    SYSTEM: ACTUATOR
      SYSTEM:  MOMENTUM_WHEEL
        STATE:  NOMINAL
            ITEM:   ACRWSP
                VALUE:          200.0
                TOLERANCE:    150.0
                DESCRIPTOR:   WHEEL_SPEED
        END_STATE
      END_SYSTEM
      SYSTEM:  TORQUER
        STATE:  NOMINAL
            ITEM:   ACTQRDPX
                VALUE:          0.0
                TOLERANCE:    20001.0
                DESCRIPTOR:   COIL_ X_DIPOLE
            ITEM:   ACTQRDPY
            ITEM:   ACTQRDPZ
            ITEM:   ATQRXI
                VALUE:  0.0
                TOLERANCE:    100.0
                DESCRIPTOR:   COIL_X-CURRENT
            ITEM:   ATQRYI
            ITEM:   ATQRZI
        END_STATE
      END_SYSTEM
  /* ACTUATOR STATE */
      STATE:  NOMINAL
          ITEM:   TORQUER_STATE
               VALUE:  NOMINAL
          ITEM:   MOMENTUM_WHEEL_STATE
               VALUE:          NOMINAL
     END_STATE
    END_SYSTEM
  /*   ACS HEALTH STATE  */
    STATE: NOMINAL_SUN
        ITEM:   SENSOR_STATE
             VALUE:  NOMINAL_SUN
        ITEM:   ACTUATOR_STATE
             VALUE:  NOMINAL
    END_STATE
    STATE: NOM_SHADOW
        ITEM:   SENSOR_STATE
             VALUE:  NOMINAL_SHADOW
         ITEM:   ACTUATOR_STATE
              VALUE:  NOMINAL
    END_STATE
  END_SYSTEM
```

References

Altair Corporation, "Applications of State Space Modeling Techniques to Launch Vehicle and Satellite Operations," Altair-CML-01-05B-DA, Bowie, Md., 1994.

Altair Corporation, "Finite State Modeling and Procedure Automation Language for Spacecraft," Altair-CML-01-05B-DA-SUPP, Bowie, Md. 1994.

Lin, David, Sue Hoge, Jim Klein, and Rex Pendley, "Use of XRunner for Automation," Proceedings of the International Telemetering Conference, San Diego, California, October, 1996.

Pendley, Rex, E. J. Scheidker, D. S. Levitt, C. R. Myers, and R. D. Werking, "Integration of a Satellite Ground Support System Based on Analysis of the Satellite Ground Domain," Paper No. SD.3.f., Proceedings of the Third International Symposium on Space Mission Operations and Ground Data Systems, Greenbelt, Md., 1994.

Scheidker, Eric, "IMACCS, A Progress Report of NASA/GSFC,s COTS-based Ground Support Systems, and Extension into New Domains," Proceedings of the International Telemetering Conference, San Diego, California, October, 1996.

Welch, B., Practical Programming in TCL and Tk, Prentice Hall, New Jersey, 1995.