Abstract for IAA International Conference on Low-Cost Planetary Missions

Automated Constraint Checking of Spacecraft Command Sequences

Joan Horvath, Leon Alkalaj, and Karl Schneider
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
Joseph Spitale
California Institute of Technology
Pasadena, CA 91125

Making certain that spacecraft command sequences do not violate any constraints is often tedious and expensive in terms of both personnel and software development. '1'0 reduce this cost, we have pursued the development of a flexible system for specifying models of spacecraft behavior in response to commands as well as constraints on that behavior. The potential need for modeling complex spacecraft behavior required that the system be designed to be usable both on a conventional workstation and a parallel supercomputer. Finally, it needed to be intuitive enough for the intended mission operations users to easily design sets of rules and models to automate tedious, resource-consuming constraint checking of commands.

We have defined a Specification And Verification Environment (SAVE) for spacecraft flight rules. The "Specification" part of the environment allows the user to specify the system behavior without specifying constraints. This behavior is represented using a finilc-slate. machine approach similar to the statecharts introduced by D. Harel. The specification can be simulated so as to validate system behavior without imposing constraints (m that behavior. The "Verification" part of SAVE allows a user to check constraints on the state space. These constraints can be behavioral, imposing some ordering (m c.vents, or the constraints can be time-dependent. The Iwo parts of SAVE collectively allow the clear specification of behavior as well as the verification of sequence integrity. A sequence of commands is interpreted as a set of events to the models. With each command the models change state according to their specification. Flight rules are expressed as constraints over the same state space as in the specification. Whenever an event occurs that changes a state variable for which a constraint exists, this constraint is checked; if the constraint is not satisfied, the user is notified.

This SAVE system was implemented in the C programming language. Initially, it was developed as a prototype running on a parallel computer. This prototype was then adapted to run (m a desktop workstation for actual flight use for TOPEX since the '1'01)EX rule set was small and did not require the computing power Of the parallel system. Interesting issues arose during this adaptation about proper testing techniques for a system of this nature; these issues will be addressed in our paper. A graphical user interface was developed that allows the specification of both models and constraints. Models are developed in a spreadsheet-like tool and rules are input using a straightforward template. Users can be trained to create rules and models in a matter of a few hours, and no programming is required. The '1'01>EX-adapted version of SAVE is currently being used by the '1'01'1 EX flight project for command sequence checking. This adaptation is in main-line night use and has replaced previous hand-checking of 12 rules (8 of them mission-critical rules) and a variety of lesser constraints. We are proposing this technology for a variety of other users both at JPL and in industry, and expect that there will be a uses of this technology for broad problems of modeling and controlling time-varying constrained critical systems. We will describe both the software system and our lessons learned in bringing this new technology to flight operations. The work described in this abstract was Jm'formed by the Jet Propulsion Laboratory, California Institute of Technology, under contract to the National Aeronautics and Space Administration.