SPACES , AN INTEGRATED SOFTWARE APPROACH FOR ATTITUDE DETERMINATION, CONTROL AND POINTING SYSTEMS ANALYSIS

James Billing-Ross*
and
Douglas Havenhill**

Honeywell Satellite Systems Operations
Glendale, Arizona

The recent success of the Pegasus small satellite launch system by Orbital Sciences Corporation foreshadows radical changes to the satellite industry comparable to those which occurred in the computer industry when personal computers became commercially available alternatives to mainframes. In order to support low cost, fixed price contracts for small commercial satellites, engineering design cycles for satellites and satellite subsystems will have to be shortened, and accomplished with fewer staff to meet more stringent cost and schedule goals. To accomplish this, better design tools must be made available which will allow the mission analysis, requirements analysis, and other systems engineering tasks to be accomplished in an integrated software environment by a systems engineer. The Satellite Pointing and Attitude Control Engineering System (*SPACES*) is a software package developed as part of an integrated toolset by Honeywell Satellite Systems Operation to meet the need for attitude determination, control and navigation subsystems requirements analysis. *SPACES* is specifically designed to support initial mission analysis, pointing and tracking system requirements, as well as ACDNS sensor and actuator analyses. The approach used in *SPACES* was to take advantage of the state-of-the-art in user interface technology to provide a integrated system preliminary design tool that is easy to use with graphically oriented output that can handle a large class of satellite missions without requiring software modification.

INTRODUCTION

The traditional approach to designing spacecraft flight control systems involves a large team of engineers, coordinated by a system engineering office, working independently on the problems in their respective disciplines. Mission analyses, orbit design, solar torque analyses, control system design and stability analyses are all performed by specialists with their own set of tools. While this method has worked fairly well for large satellite development efforts funded with cost-plus type contracts, it has many disadvantages when applied to smaller satellites designed for commercial or experimental programs funded under fixed price contracts with aggressive development schedules. These disadvantages include slow communication, excessive bureaucracy and paperwork, and high cost. The resource and schedule constraints of a commercial lightsat program cannot, in general, be met using the traditional approach, so significant improvements to current methods of designing and building spacecraft are required.

In order to meet tight delivery schedules on fixed price contracts, the design cycle must be accelerated such that a design can be effectively finalized at the time the proposal is submitted. The contract then becomes a matter of systems integration and test rather

---

* Staff Engineer, ACDNS Systems Engineering, Member AIAA

** Engineering Section Head, Mechanisms and Pointing Systems , Member AAS

than development. Due to constraints of time and resources , a smaller multi-disciplinary team with strong systems orientation working closely together must replace the larger traditional staff of specialists and systems engineers. In order for this team to be successful, new integrated design tools must be developed to maximize the productivity of the design staff. These tools should use less expensive computer resources than current programs which use costly mainframe time and usually have steep learning curves and require extensive processing of results in order to generate usable reports. The new generation of general purpose computer aided engineering (CAE) tools available to support spacecraft flight control system design have improved matters considerably. Through the use of improved graphical interface technology, the development of commercial packages for controls and mission analysis have made the design team's task easier. Unfortunately, to be commercially viable, these packages are developed for as broad a set of applications as possible, and ,as a result , do not support a multi-disciplinary approach. As an example, currently available control systems CAE tools provide effective means to design and analyze control systems, but do not address the issue of mission requirements for a specific satellite mission. Commercial software tools are available which allow detailed orbital analyses and mission analyses, but do not address the pointing and attitude control requirements to meet the desired mission objectives. The development of new software for spacecraft design must bridge this gap in order to provide the necessary tools for commercial lightsat work.

DEVELOPMENT OF AN INTEGRATED TOOLSET

The design cycle of a satellite ACDNS subsystem traditionally goes through several protracted stages after the initial contract has been awarded. A preliminary design concept is developed for the proposal with first pass analyses for backup. Once the contract is won, a different design team may take over and start redesigning based on new requirements and proceeding through Preliminary Design Reviews (PDR) and Comprehensive Design Review (CDR). If the contract is a cost plus type contract, this cycle will provide successful results given enough time and money. A fixed price contract has radically different ground rules in that, unless the design is mature when the initial contract is negotiated, the contractor is taking on considerable (and usually unacceptable) financial risk. More than one contractor has been burned by trying to develop new systems with success-oriented schedules and fixed funding.

To avoid the risk associated with a fixed price contract, the traditional preproposal design cycle should be accelerated such that a mature product design based on a well understood technology base can be proposed instead of a preliminary concept. In addition, the hardware components of the system should have been previously designed and developed such that the system can be integrated based on the selected configuration.

In order to provide rapid development capability for spacecraft flight management systems, engineers at Honeywell have initiated the *Proteus A* project to develop a subsystem approach for lightweight satellites based on integrated design tools and modular hardware and software. The core of the flight management, and the current focus of the project team is a mission adaptable, modular attitude determination, navigation, and control system. The accelerated design cycle for supporting fixed price commercial type ACDNS systems is shown in Fig. 1.

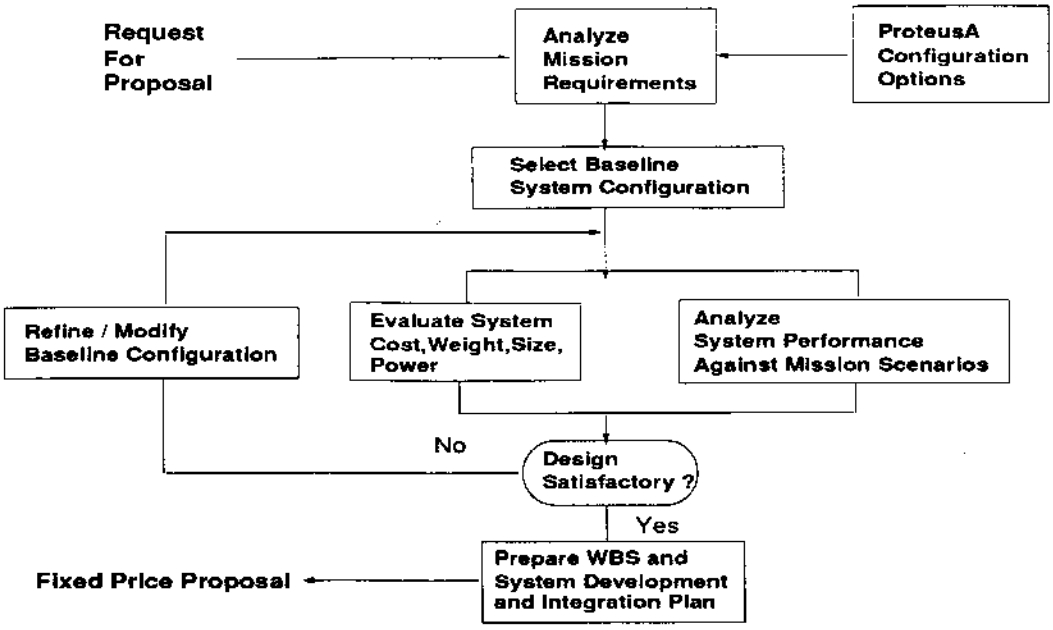


Fig. 1 *Proteus A* Accelerated Design Cycle

One of the goals of the *Proteus* team is to refine the design process so that there is sufficient time to iterate several times on the proposed configuration in order to refine the design and locate any flaws. Since the amount of time allowed for the proposal process is generally counted in weeks, the designers must be supported by design and analysis tools which allow for rapid analysis and prototyping. The focus of the tools development by the *Proteus A* team has been to develop an integrated toolset for performing the requirements analysis and performance verification tasks as shown in Fig.2.

Requirements Analysis

- Mission Design / Analysis
  - ground station visibility
  - revisit frequency
  - ground coverage
- Disturbance/Maneuver Torque Requirements
  - momentum/torque envelopes
- Orbit Stationkeeping Requirements
- Tracking Requirements
  - geolocation tracking
  - intersatellite tracking
  - sun/moon/celestial object tracking (angle,range,range rate)
- Navigation requirements

ACDNS Configuration Analysis

- Sensor Requirements/Placement Analysis
  - Star Cameras
  - Horizon Sensors
  - Inertial Rate Sensors
  - accelerometers
- Actuator Analysis
  - momentum control device configuration
  - magnetic torquer sizing
  - reaction jet ISP/fuel load analysis
- Vehicle Mass Properties Analysis

Performance Analysis

- Non-linear systems time domain simulation
- Linear systems analysis frequency/time domain
- Attitude determination/ navigation system covariance analysis

Fig. 2 Integrated Tool set Requirements For Accelerated ACDNS Design Cycle

Since the team developing the *Proteus* software tools is also performing the ACDNS advanced development work ,the top priorities for meeting different task requirements are driven by the current spacecraft design requirements with additional tasks completed as time and resources allow.

In order to take advantage of existing CAE tool development at Honeywell, the design toolset requirements are divided among four separate software applications (see Fig. 3) using a Apple Macintosh II computer as the common platform.

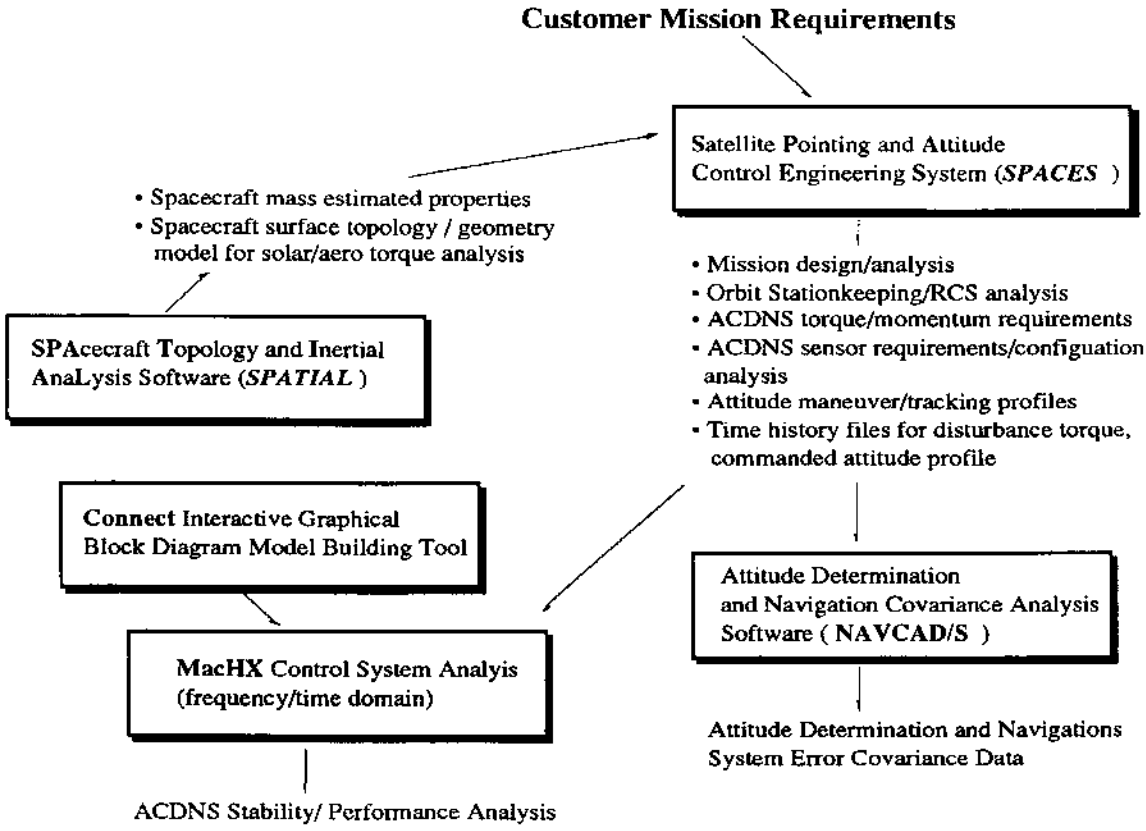


Fig. 3 *Proteus* Integrated ACDNS Toolset

The focus of the software work at the Satellite Systems Operation in Glendale has been on the development of the *SPACES* program for ACDNS requirements analysis due to current requirements to support both advanced lightsat design studies and preproposal work. NAVCAD, McHX and CONNECT were existing Honeywell CAE tools developed at the Systems and Research Center which were able to be adapted for space applications with minor or no modifications required. The overall direction and goal of the *Proteus* effort is to develop the tools in direct response to needs of the satellite ACDNS designers on current programs and proposals. Since the team developing the software tools is also performing the engineering work, this tends to keep the software development focused on actual design requirements and minimizes unnecessary bells and whistles.

DESIGN AND IMPLEMENTATION OF *SPACES*

The *SPACES* application software is the successor to software originally developed for pointing system preliminary design studies. Since these studies started with mission design and orbit selection, and proceeded all the way through controls and mechanical analysis of the concept design, the original software evolved to address all of these design

areas. The major drawback to the original software was that ,like many traditional engineering software tools, it had to be rewritten constantly to handle different problems, the user interface was cumbersome and the overall package was difficult for anyone else to learn to use quickly.

The development of *SPACES* began as an off-hours project to extend the approach used in the original software to provide a general design tool for ACDNS and pointing systems analysis. The initial goal was to rewrite the original BASIC software into a full Macintosh application using Pascal with multiple graphics windows, pull-down menus and following the Apple Human Interface Guidelines. This style of programming is a departure from the usual engineering tools development as the interface is a central part of the software design instead of an afterthought. A major difference that the authors encountered when restructuring the original BASIC code, is that Macintosh software is designed based on a continuously running event loop architecture, interrupt driven by user actions such as mouse and keyboard commands as opposed to the general batch design that was used previously. In order to accommodate this architecture, as well as the variety of analysis options that were envisioned, extensive use of structured data objects and library modules was adopted early in the project. The use of an integrated software development environment (THINK Pascal™) also allowed for the management of multiple libraries and encouraged good software practices. As a result of these choices, as well as the maturity of the Macintosh operating system toolbox routines and development tools, the first version of *SPACES* was implemented fairly quickly , and in time to provide significant support for a satellite ACDS proposal.

As a detailed discussion of the software design and implementation is beyond the scope of this paper, the description of the software implementation in the following sections will focus on some of the key structural and design elements of *SPACES*.

Software Architecture

The basic architecture of *SPACES* is designed externally to provide the user maximum flexibility in analyzing a problem and internally to provide for additional analysis modules that may be required. *SPACES* uses drop-down menus including the standard Apple, File , and Edit Menus, as well as several application specific headings. The *SPACES* menu options are grouped to support the basic tasks which must be accomplished in order to run an analysis and can be selected in any order. These include defining an orbit (at least one, but up to seven at present) , setting the simulation parameters, and selecting an output analysis display. In addition there are a series of options for specifying the environmental models and analysis suboptions. This allows a simulation run to be modified with different orbit parameters or different environmental effects as desired by the user. Multiple windows are supported so that results of different runs can be compared on the screen. Extensive internal checking is included to minimize invalid results from improperly setting up an analysis. For example, magnetic torque analyses can not be run without first activating the magnetic field model.

The internal structure of the *SPACES* software is based on an event loop which is a continuous running interrupt driven loop which senses and interprets system events, and passes the information back to the program to be handled. If a mouse click occurs within a menu area on the screen, or a menu command key combination occurs, the menuhandler sorts out the actions to be taken based on the menu item selected. Most of the basic functions are standard Macintosh algorithms which simplified the development process. The *SPACES* specific procedures are accessed from the menu structure as shown in Fig. 4. Most of the menu items shown function to set up the specific conditions of the analysis run.

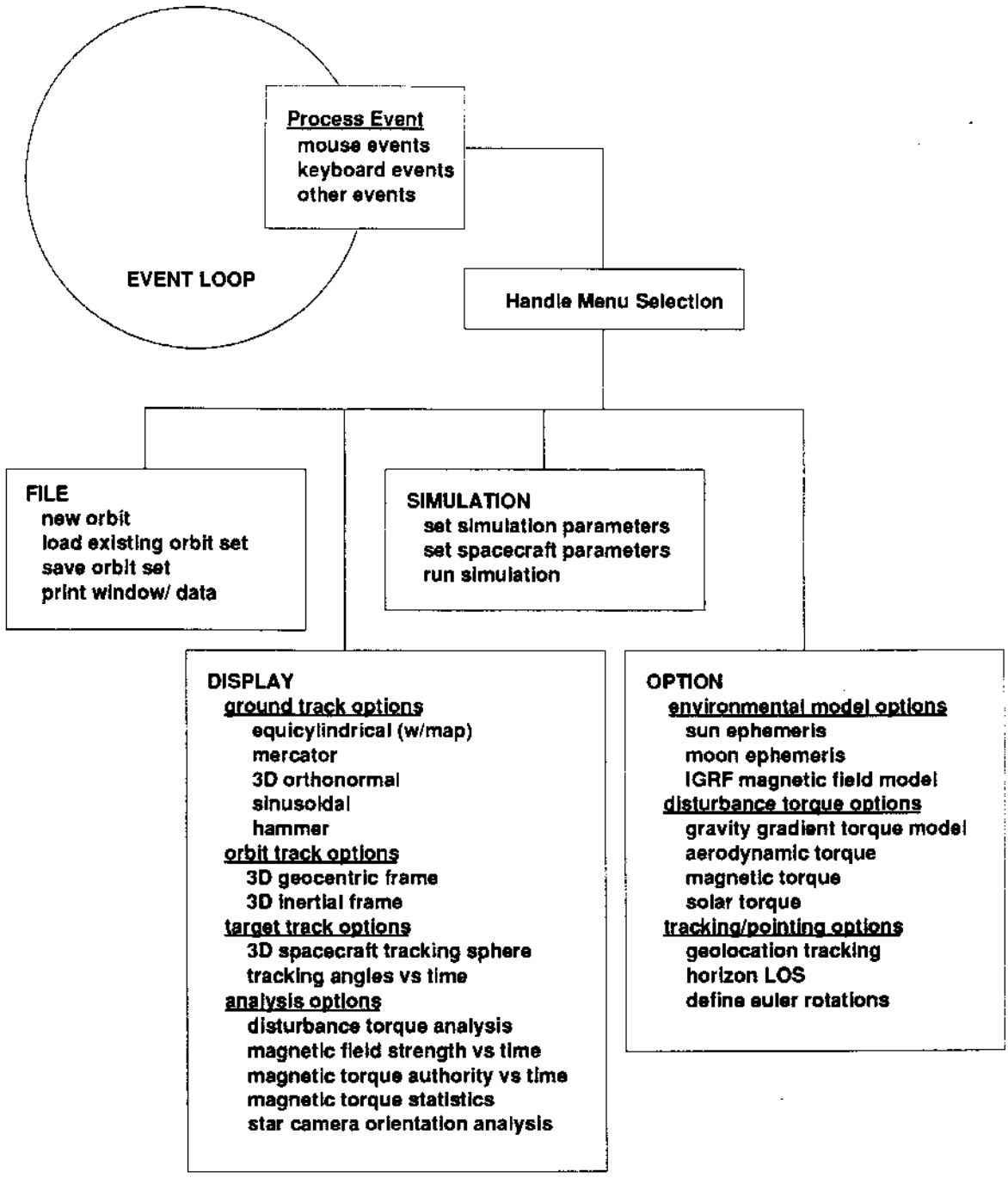


Fig. 4 *SPACES* Event Loop And Menu Structure

The heart of *SPACES* analysis capability is a time domain simulation which propagates the orbits and performs the analyses at each step in the orbit. As the satellites move around their orbits, analyses are performed to determine the satellite environment, the relative location and rates of the satellites as well as the position of the satellites with respect to the earth. The basic simulation loops and tasks are shown in Fig.4.

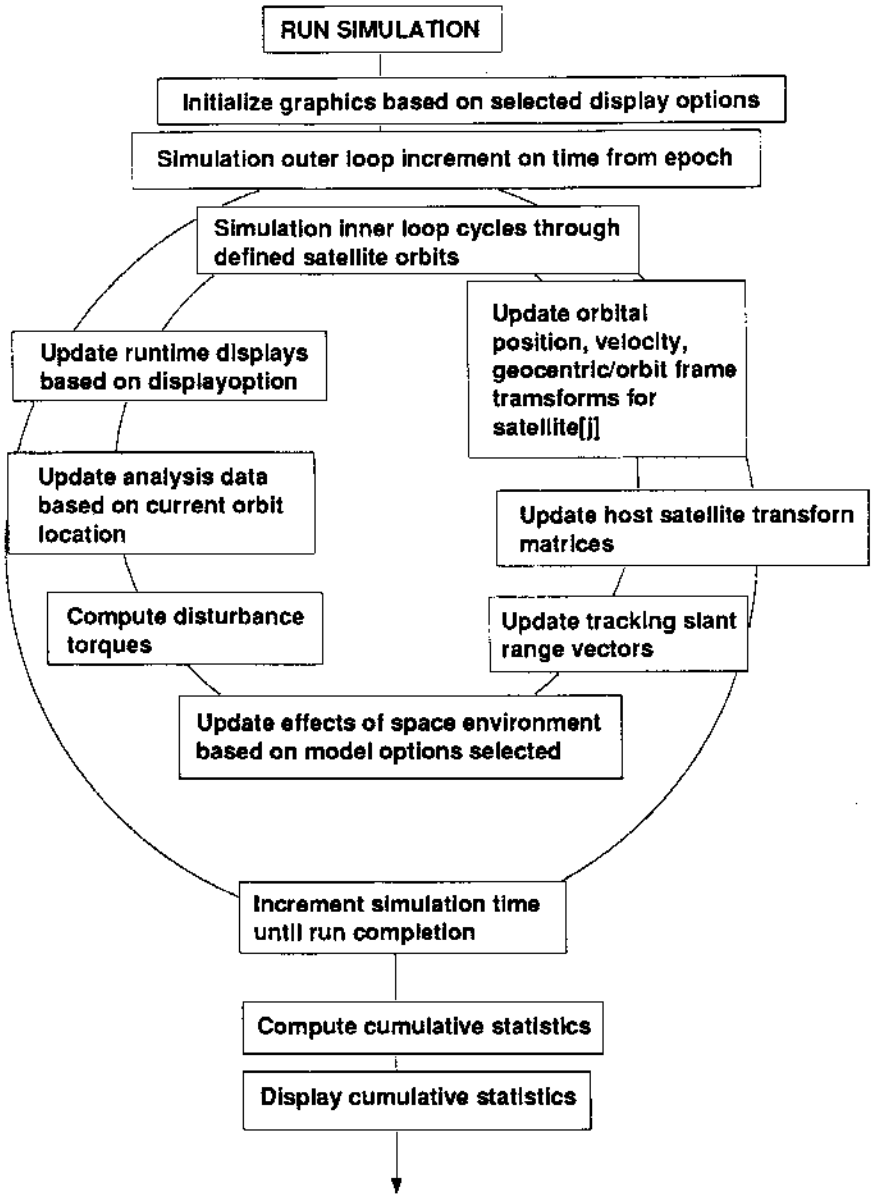


Fig. 5 *SPACES* Simulation Loop Structure

In order to support multiple satellites and targets*** as well as multiple graphics displays, extensive use was made of structured data objects. This made it relatively simple to keep track of multiple orbits and tracking vectors as well as the attributes of the various displays (see Fig. 6).

***The term *target* is used in SPACES to describe anything you want to point at, and does not have any specific weapons systems connotations.

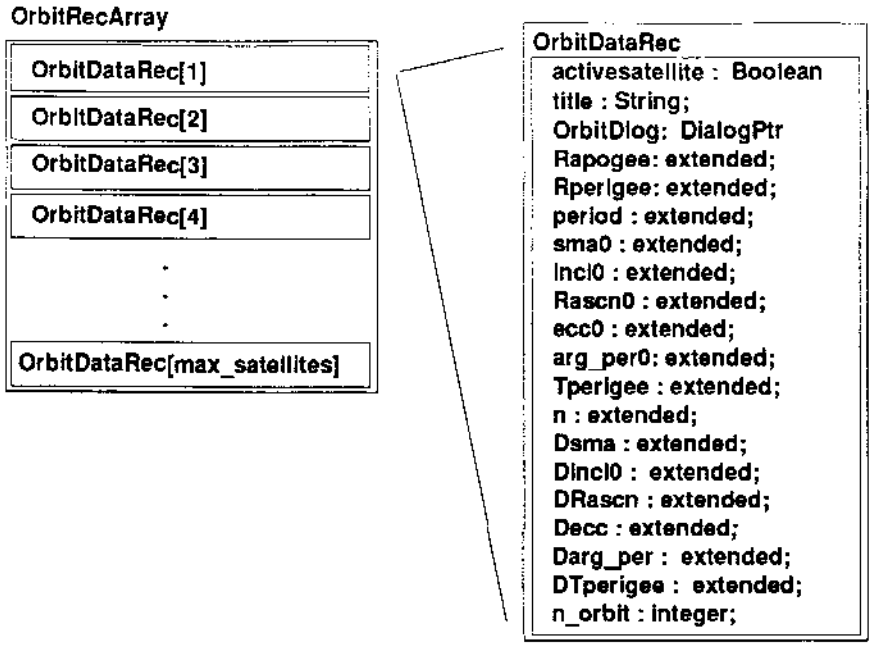


Fig. 6 Data Structure For Handling Multiple Satellite Orbits

Most of the data structures in *SPACES* have been implemented initially as static arrays of records for convenience. As the program develops and memory usage becomes critical, more use will be made of dynamic data structures which will only occupy the space required for the case being studied. This became a problem during the implementation of the star camera orientation analysis option, where the search grid data array had to be sized very large to handle the minimum grid spacing of 1 degree. This was resolved by implementing the search data array as a link list of data records which could be expanded dynamically to meet the actual grid sizing.

Getting Data Into The Program

The task of getting data into a program such as *SPACES* usually involves a lengthy list of questions which require responses from the user. This can involve significant time on the part of the user to respond to all of the questions, many of which may not be significant to the case being studied. This can be especially tedious when many runs are being made, and as a result, the alternate method is to set up 'batch' input files that can be edited for a specific case. While this can be a satisfactory approach, it involves the maintenance of the input file as well as the program itself. This problem is handled in Macintosh software generally and in *SPACES* specifically through the use of pull-down menus and dialog boxes. Dialog boxes contain a description of the data involved, as well as editable text regions, control buttons, scroll bars, and other user defined input objects. Once the dialog box is called by a menu selection, the user can take advantage of default settings, or change the data and settings as required. Once the user is satisfied, he or she can click on the OK button and proceed onward. This is made even easier by the use of the resource editor which can be used to create and edit the dialog box templates. The resource editor allows the software developer to graphically design the dialog box for maximum usefulness, and allows the user to customize the default setting in the finished application. In addition, the dialog boxes can be called up at any time in the program and edited to change the selections and data.

SPACES currently uses five dialog boxes to accept input data from the user to define the satellite orbits, set up the simulation run and specify spacecraft mass properties, star search criteria, and geolocation tracking sites. Two of the dialog boxes must be opened to

set up a run. These are the orbit data dialog (see Fig. 7) and the simulation parameter dialog(see Fig. 8).

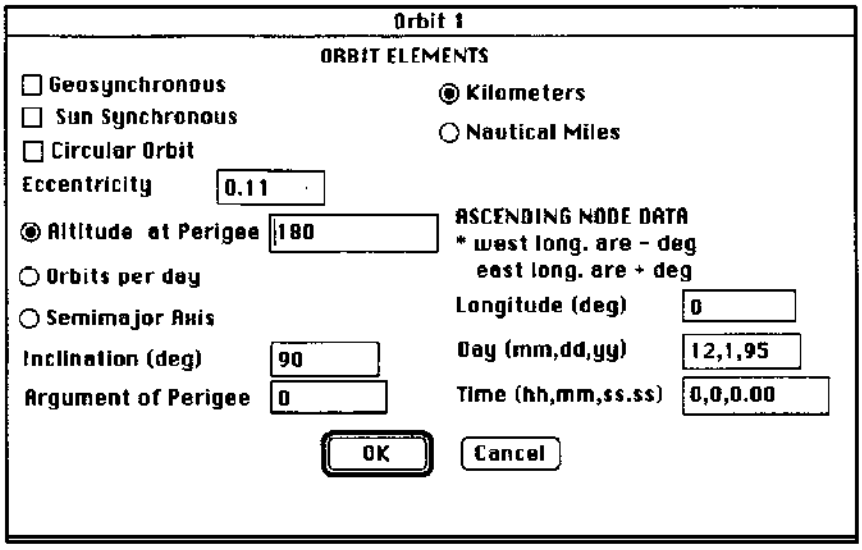


Fig. 7 Orbit Data Dialog

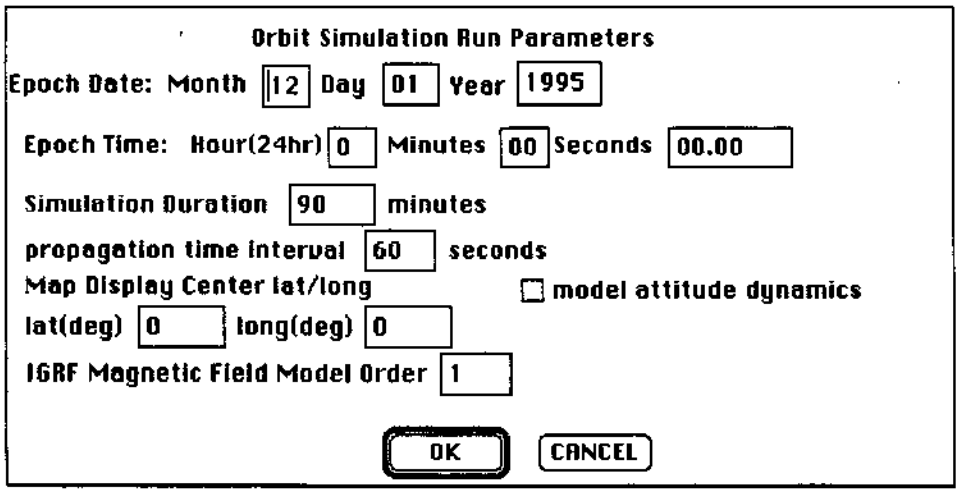


Fig. 8 Simulation Parameter Dialog

The orbit data dialog is used to define an orbit and is setup to allow quick specification of an orbit for design purposes. A minimum of one orbit must be defined and as many as seven are handled in the current version. The simulation parameter dialog sets up the overall run specifications and specifies the look angle for the displays. These settings as well as the orbit parameters can be changed at any point. can be changes. The default parameters for each dialog box can be customized by editing the dialog resource. Other dialog boxes are used to set up the star camera analysis, specify spacecraft properties and control capabilities, and specify geolocation tracking sites (see Fig. 9).

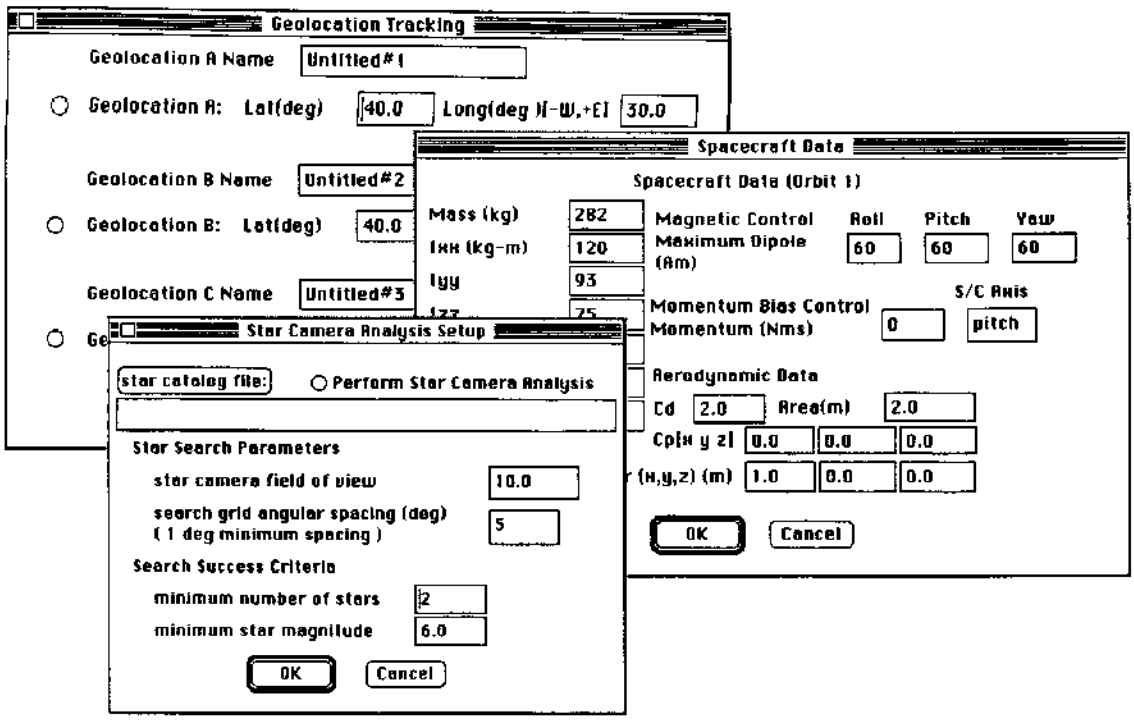


Fig. 9 Dialog Boxes Used For Data Input

Analyzing The Results

SPACES makes extensive use of graphical analysis displays to integrate as much information as possible in a meaningful fashion. The ability to create and maintain multiple windows also allows comparative studies to be performed and analyzed. In order to facilitate the use of the display data, the windows can be printed directly or copied to the clipboard and pasted into other Macintosh applications allowing viewgraphs and reports to be quickly generated.

The heart of the display processing is a library of projection algorithms which allow three-dimensional coordinates to be translated into a two-dimensional reference frame. The plot library then converts the two dimensional coordinates in a user coordinate frame to the integer coordinates used by the Macintosh QuickDraw library. Although the primary usage of these routines is to generate maps showing the satellite ground track, the projections are used for spherical orbit track plots, target tracking plots , as well as for analyzing optimal star camera orientation. An example of the use of two-dimensional and three-dimensional displays is shown in Fig. 10.

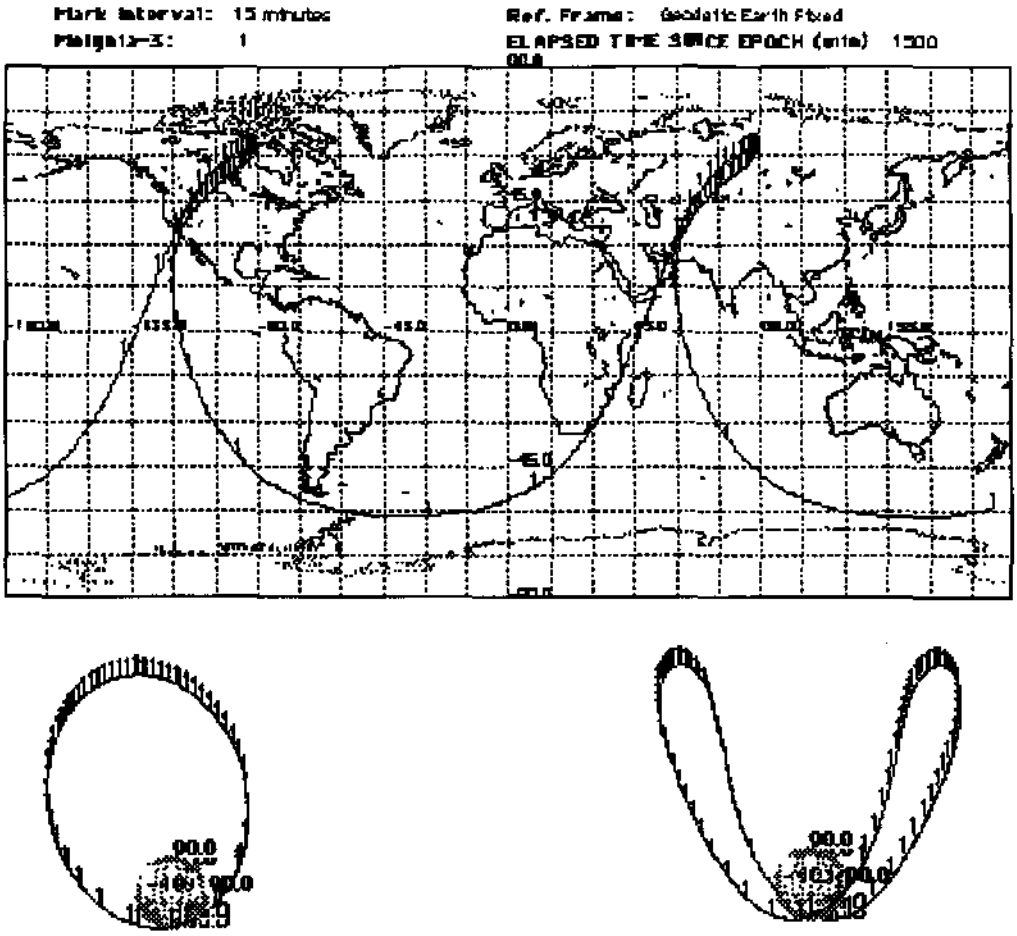


Fig. 10 Two and Three Dimensional Analysis Displays For Molynia-3 Satellite Orbit

The spherical projection format is also used for displaying target track information and provides information on track angles, visibility, as well as earth horizon at apogee and perigee. An example target track display is shown in Fig. 11

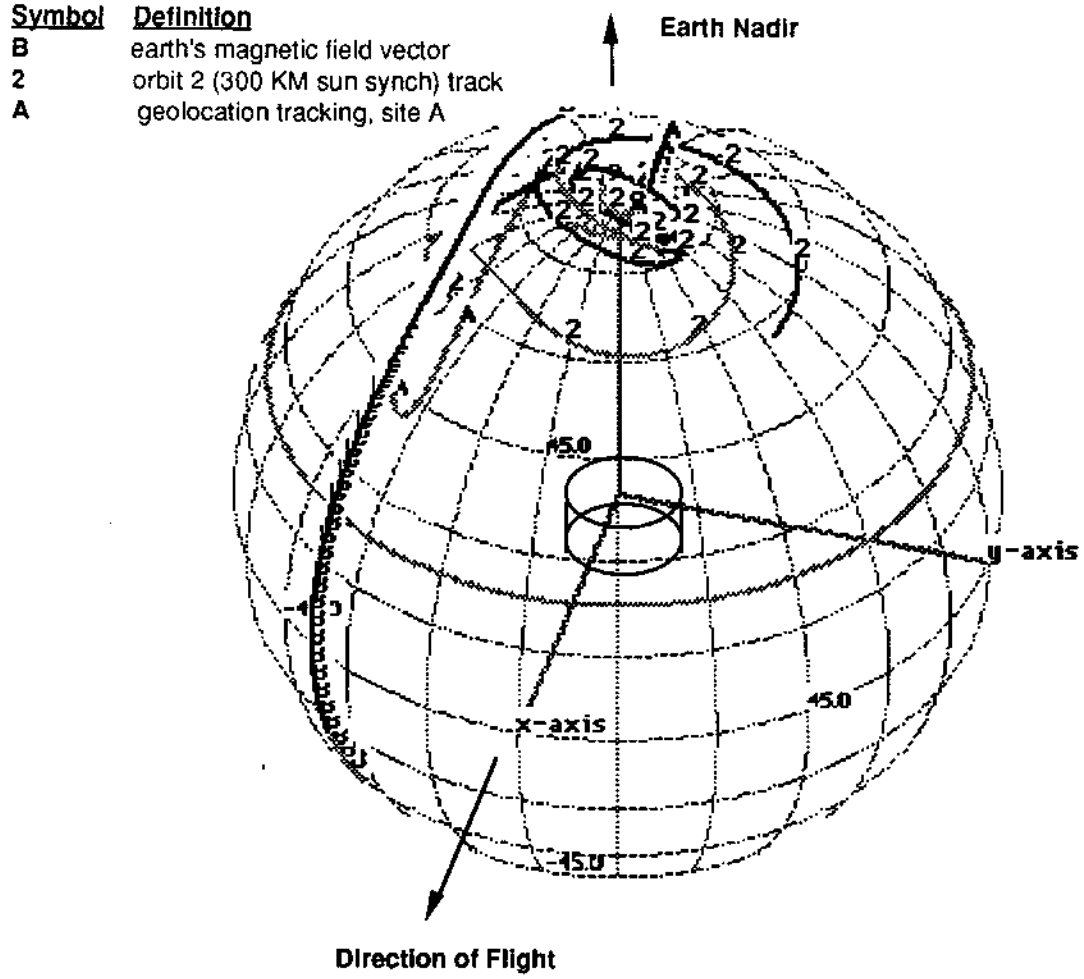
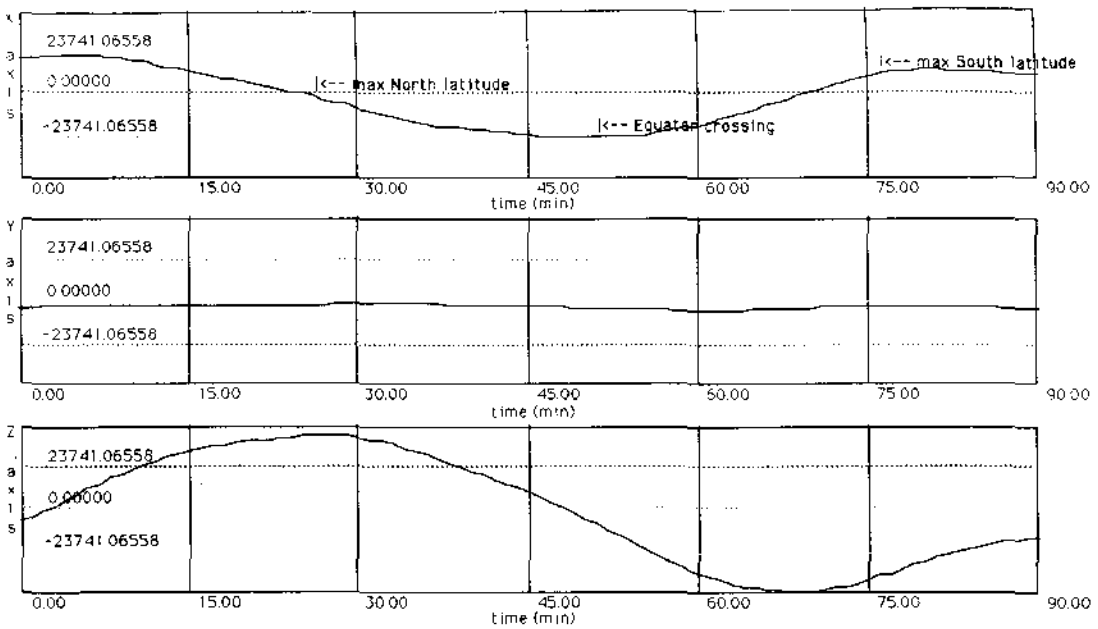


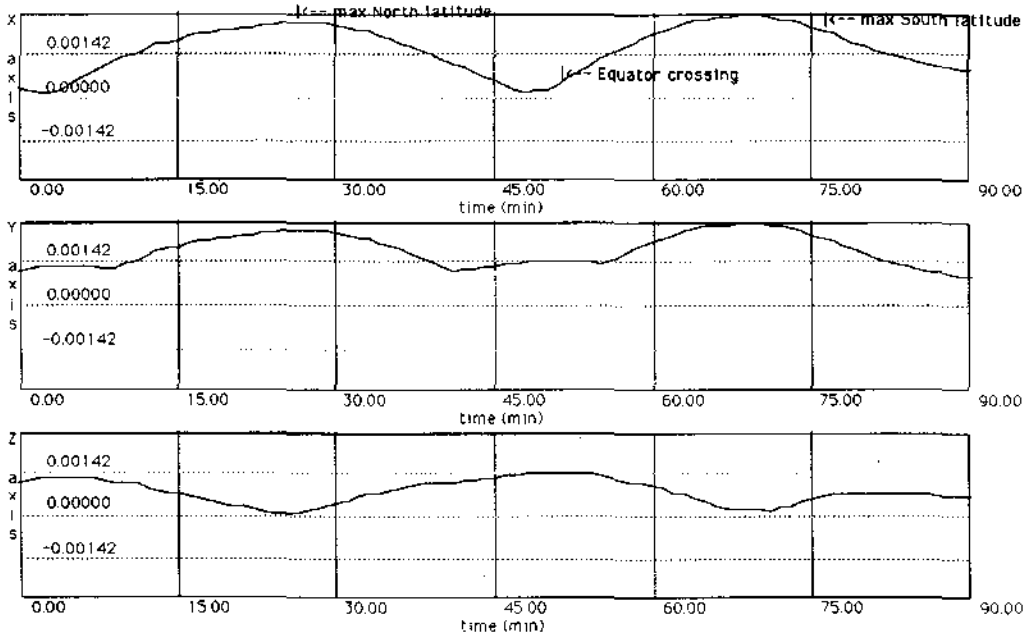
Fig. 11 Three Dimensional Tracking Profile Display

SPACES also uses more traditional strip chart formats for displaying time history graphs. The control capability of a system employing magnetic control can be analyzed by evaluating the magnetic field strength (Fig. 13), sizing the magnetic torque rods and running torque capability time histories (Fig. 14) and cumulative statistics (Fig. 15) to show the probability of achieved desired torque levels in each axis.



Magnetic Field Strength In Spacecraft Coordinates For 413nm Polar Orbit
(based on 8th order International Geophysical Reference Field Model)

Fig. 13 Magnetic Field Strength Analysis



Magnetic Torque Capability In Spacecraft Coordinates Based On 60 Am Torque Bars

Fig. 14 Magnetic Torque Authority Analysis

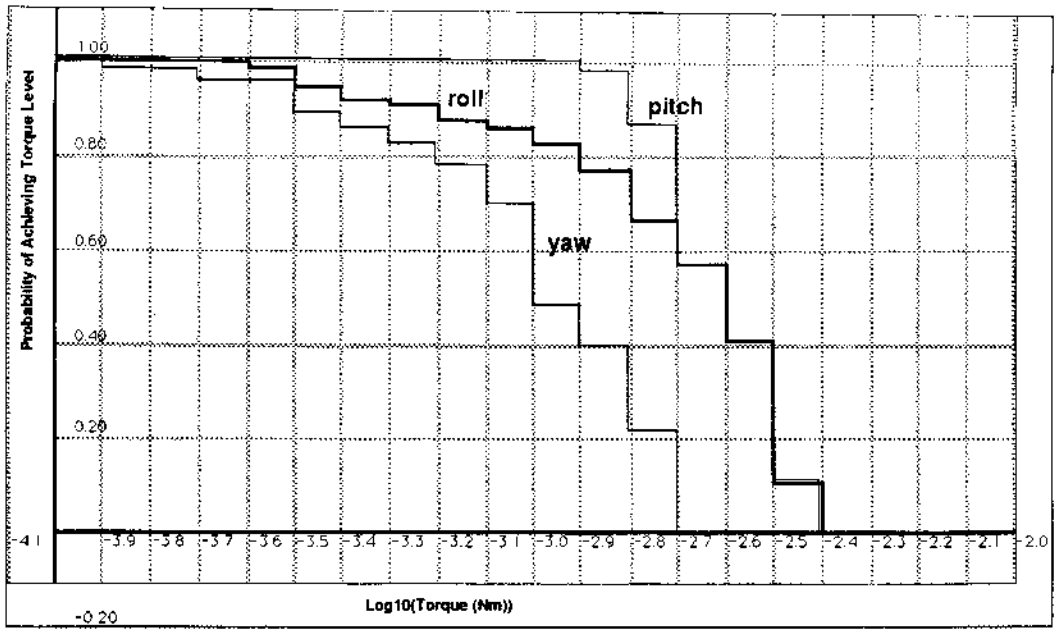


Fig. 15 Cumulative Probability Analysis For Magnetic Controller

APPLICATIONS AND FUTURE DEVELOPMENTS OF *SPACES*

SPACES has been an experiment in integrating mission analysis and attitude and pointing system tools and has proven to be very useful in preliminary design and preproposal work at Honeywell. *SPACES* has been used in supporting preproposal design efforts as well as ongoing ACDNS system design studies, and is evolving the meet the needs of these programs. Current development work is focused on supporting star tracker analyses and gimbal pointing system studies. The companion software for specifying complex vehicle topologies, *SPATIAL,* is being developed this year under IR&D funding and the goal is to demonstrate processing with *SPACES* later this year.

The success of *SPACES* has pointed out the important role for design tools which can support multi-disciplinary analyses. The goal of the *Proteus* project team is to develop these tools and methods , as well as the associated modular flight hardware and software, in order to provide a flight management subsystem solution for commercial lightsat applications.