N90-25511

StarPlan: A Model-based Diagnostic System for Spacecraft

Dennis Heher Paul Pownall

Artificial Intelligence Group
Ford Aerospace Corporation
1260 Crossman Avenue
Sunnyvale, California 94089

ABSTRACT

The Sunnyvale Division of Ford Aerospace has created a model-based reasoning capability for diagnosing faults in space systems. The approach employs reasoning about a model of the domain (as it is designed to operate) to explain differences between expected and actual telemetry; i.e., to identify the root cause of the discrepancy (at an appropriate level of detail) and determine necessary corrective action. A development environment, named Paragon, has been implemented to support both model-building and reasoning. The major benefit of the model-based approach is the capability for the intelligent system to handle faults that were not anticipated by a human expert.

The feasibility of this approach for diagnosing problems in a spacecraft has been demonstrated in a prototype system, named StarPlan. Reasoning modules within StarPlan detect anomalous telemetry, establish goals for returning the telemetry to nominal values, and create a command plan for attaining the goals. Before commands are implemented, their effects are simulated to assure convergence toward the goal. After the commands are issued, the telemetry is monitored to assure that the plan is successful. These features of StarPlan, along with associated concerns, issues and future directions, are discussed in this paper.

INTRODUCTION

The satellite network of the United States is a strategic resource which requires continuous monitoring and maintenance to ensure it supports defense requirements. System support personnel must carefully and precisely monitor and command individual satellites to sustain the satellite's readiness.

In current operations, when anomalies occur, a carefully developed process of evaluation, testing, diagnosis, and planning is executed by a team of highly trained engineers which support each satellite system. This process is applied incrementally to safe the vehicle, isolate the source of the problem, resolve the anomaly, and continue operations. Later, this process is permanently recorded as a contingency procedure and utilized whenever similar conditions reoccur.

Ford Aerospace Corporation, Sunnyvale Division, has been working in the field of Artificial Intelligence since the early 1980's developing a system called Paragon which, when given the proper functional description of a satellite, can monitor telemetry data, notice anomalous conditions, and recommend corrective actions.

PARAGON

Paragon is one of Ford Aerospace's innovative development environments for building model-based "intelligent" systems. It is an unusually effective software and interface system, which allows the user to go directly from idea to implementation simply by describing domain components and their behavior with logical or mathematical functions. In most cases, these can be entered simply by mouse selection within a structured window and menu driven interface. Paragon allows an expert to transfer his mental model of the domain to the computer without being taxed by normal coding and software development procedures.

Knowledge Base Development

Paragon provides automated knowledge acquisition aids that interact with an expert system developer to build a knowledge base that is a model of the problem domain. The developer is given design freedom to model a domain in a way that is most natural to his or her application.

The model consists of concepts (physical or non-physical objects) that comprise the domain, appropriate characteristics of the objects (e.g., height, weight, color, current, voltage, etc.), the interaction or relationships with other domain concepts (e.g., electrically connected to, supplied by, etc.), the behavior of the concept such as the states in which it exists (e.g., ON, OFF, IDLE, etc.), what events occur while in each state, and what causes the concept to transition from one state to another.

The model is developed via a graphic interface using pop-up menus and mouse selection. The use of typing is limited to assigning names to concepts, states, etc. Once a name has been assigned, it appears in menus or graphic displays for subsequent selection.

As the model is being developed, Paragon collects the information and automatically translates it to a representation designed for inference and problem solving. A simulator option is provided that automatically generates software code so that the behavior can be simulated and parameters displayed for verification by the system developer.

Concepts can be conceptual or physical objects (or components) that have specific meaning, relationships, and behavior in the domain. For example, in the Electrical Power Subsystem (EPS) of a satellite some of the components would be +Y WING, -Y WING, BATTERY 1, BATTERY 2, and BATTERY 3. Once the concepts are decided upon, the developer creates a classification definition. For example, BATTERY 1, BATTERY 2, and BATTERY 3 belong to the general class named BATTERIES (see Figure 1). When classification is complete, the developer designates composition relationships. The specification of concept attributes and functional relationships follow.

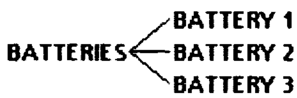

Figure 1. Class and Instance classification example.

Each concept has attributes that, once defined, allow the developer to (1) localize all characteristics and behavior of an object and (2) specify functional relationships between objects. The telemetry measurements can be attributes of specific concepts which relate to components on the vehicle. For example, the attributes for the +Y WING and -Y WING would be CURRENT and SUN ORIENTATION. Any characteristics of a component or object can be specified as an attribute.

Once attributes are defined, their value class is specified. A value class designation indicates what type, or class, of values a particular attribute may take on. For example, an attribute indicating whether a component was on or off would have an ON/OFF value class type. This type would differ from the temperature of a battery, which would be a numerical value. At this point attributes can be used when specifying functional relationships between concepts and when specifying concept behavior.

Functioı.al relationships allow the developer to specify relationships between objects or components. Figure 2 displays an example of relationships between the +Y WING and other objects in the model. The "causes" window displays values which are passed to the +Y WING and the "effects" window displays those values which are passed from the +Y WING. Each functional relationship includes a value class specification and only an attribute with the same value class as the functional relationship can be passed by that relationship. This prohibits the developer from accidentally passing, for example, an ON/OFF value when a numerical value is required.

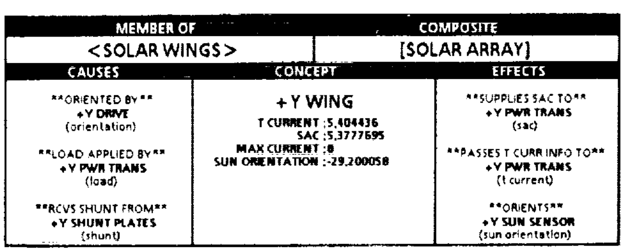

Figure 2. An example of Relationships.

Concept behavior is specified by defining (1) the states in which concepts can exist, (2) the transition conditions which determine when concepts leave one state and enter another, and (3) the attribute events which may occur in each state. Transition conditions are specified in the form of a logical operation with equations, and attribute events are specified in the form of equations.

Once concept behavior has been specified, Paragon has a "simulator" option that allows the developer to test and verify the modeled behavior. Simulations can be done at the single concept level or at the full knowledge base level. The developer is given a large amount of freedom for building simulation displays. A display can be designed that best fits the nature of the behavior to be tested or demonstrated. Display options include dials, strip charts, simple values, and flashing alarms.

Paragon's Reasoning Modules

Once a domain expert has finished building a knowledge base, Paragon can reason intelligently about the behavior as described in the knowledge base. Paragon has a collection of reasoning modules which can spot anomalous or unexpected attribute values, assess the situation and generate a list of components that could be involved with the anomaly, generate goals to correct the anomaly, and then develop a plan which will satisfy the goals.

Paragon's Data Monitoring module continually monitors the value of each attribute and when a value which is outside normal expectations is noticed, an alarm is raised. The monitoring is based upon notifications which are statements attached to concepts that specify conditions which can activate the intelligent system. Paragon's Data Monitoring module continually examines whether the current value of each attribute "matches" the defined notification condition.

Once notification occurs, the Situation Assessment module generates a ranked list of components which could have participated in causing the notification. The ranking is a "focusing" mechanism based upon the functional relationships defined within the knowledge base. With this assessment list, Paragon's reasoning modules have a significantly narrowed search space in which to find a solution to the anomaly.

With the results of the Data Monitoring module and the Situation Assessment list, the Goal Determination module identifies a change in condition (a goal or goals) which would return an out-of-limits component to nominal behavior.

Using the highest ranked component(s) identified in Situation Assessment and the goal(s) associated with that component generated from the Goal Determination module, the Planning module searches for events which have the potential to achieve the goal(s). This search is a traversal of the knowledge base across functional relationships and events that indicate, by convergence, that they would satisfy the goal(s) are identified. The transition conditions that cause these events are searched for the commands or actions which enable these events to occur.

Upon finding a plan to satisfy the given goal(s), the Planning module recommends the plan and awaits a response. If the plan is executed, the Planning module monitors the attribute values to see if indeed they do return to nominal ranges.

In order for the intelligent system to accurately confirm its operating hypothesis, the design of the knowledge base must accurately reflect the satellite command and control functionality.

The Planning module completes anomaly resolution when all goals which have been developed are achieved or, in the case of serious system failures, they cannot be achieved.

STARPLAN

StarPlan is a prototype system built with Paragon which monitors conditions onboard the Electrical Power Subsystem of a satellite, identify and diagnose problems, and advise the operator on how best to continue operations.

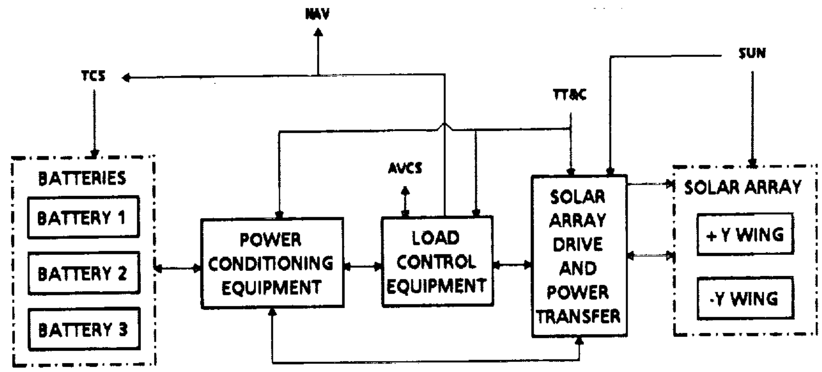

Figure 3. Functional diagram of the EPS.

The user of StarPlan would continue to control the health and status decisions concerning the satellite, but instead of asking experts to analyze the situation, the operator would simply review the recommendations of the intelligent system, making queries for additional information when necessary, and approving actions which implement the best available alternative for resolving the anomaly. With this system, the analysis, planning, and resulting command sequences are developed by StarPlan rather than by a team of satellite experts.

StarPlan Design

StarPlan consists of two knowledge bases: the first being a functional model of the EPS, and the second knowledge base a simulation model of the EPS. The functional model is a replication of the components and the relationships among those components of the EPS to provide the essential knowledge for the intelligent system to properly reason about a satellite. You could think of this knowledge base as a machine representation of a true-to-life physical model of the system.

Figure 3 is a functional diagram of the EPS and Figure 4 depicts the design of the composition of the EPS knowledge base.

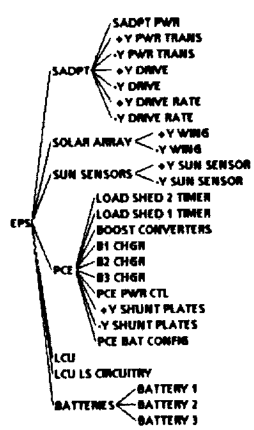

Figure 4. Composition of the EPS knowledge base.

The primary consideration in developing the knowledge base was the desire to accurately reflect the design of the actual satellite to the level of equipment configurations and functionality. When initially developing the knowledge base, the designers attempted to group too much behavior in top level components. This forced the designers to try to capture behavior that is so diverse that one did not get an intuitive "feel" for the model. The final design used subcomponents where the behavior specifications were still complex, but much more understandable.

The second knowledge base, replacing the actual satellite, is used only as a simulation model; to generate telemetry necessary to test the intelligent system. To test the intelligent system, the designers have modified this model in such a way that faults can be simulated. The faults added to this model include:

- BAD SUN SENSOR: A solar wing is unable to track the sun due to a zero error being returned by a failed sun sensor.
- WING DRIVE POWER FAILURE: A solar wing is unable to track the sun due to a System A power source failure.
- WING DRIVE ELECTROSTATIC DISCHARGE: A solar wing is unable to track the sun due to a anomalous logic change placing the wing in the hold mode.
- WING TRACKING CIRCUITRY FAILURE: A solar wing is unable to track the sun due to a tracking circuitry failure.
- BATTERY 3 THERMAL COVER DEGRADATION: Battery 3 overheats due to thermal cover degradation and a high sun incidence angle.
- BATTERY 3 HEATER THERMOSTAT FAILURE: Battery 3 overheats due to thermostat failure in the A string battery heater.
- LOAD SHED 1 TIMER FAILURE: The load shed 1 timer begins timing out independent of normal system control.

StarPlan Demonstration

The following is a description of the sequence of events during which the +Y Wing Drive Power Failure anomaly is resolved.

The satellite ground station acquires the satellite and begins to process the health and status telemetry data. Monitoring the telemetry data, StarPlan notices that several data points are out of range. Figure 5 displays the EPS telemetry data, with those values that are out of limits being highlighted.

From the notifications, the Situation Assessment module generates a list of potential components involved in the anomaly. Figure 6 displays which components could be involved with this

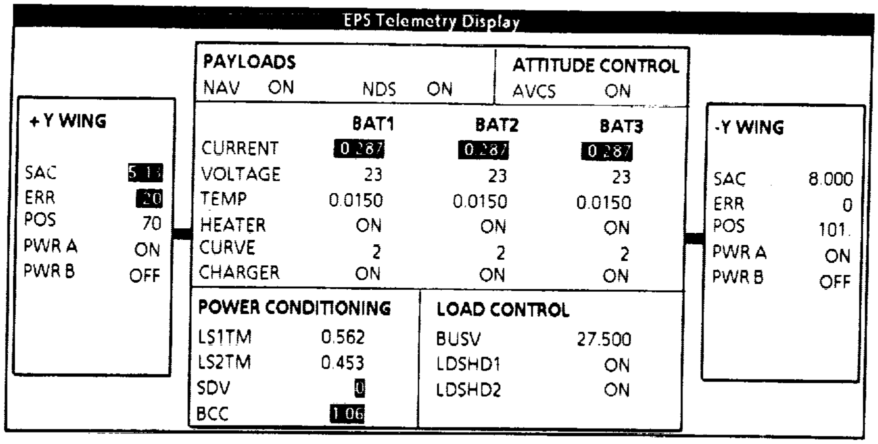

Figure 5. EPS anomalous telemetry data.

anomaly. Notice that the top ranked components are all related to the +Y WING, thus narrowing the search space for the other reasoning modules.

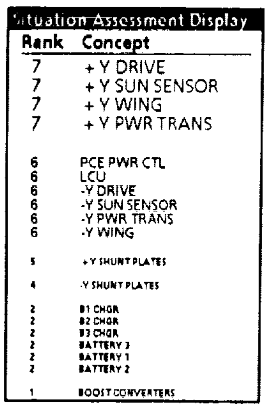

Figure 6. Situation Assessment List.

The Goals display is shown in Figure 7. The top two goals (there are actually seven goals, but only the top two are shown) are related to the highest ranked components in the assessment list. The goals, displayed in an English-like syntax for easy understanding, are essentially saying that the +Y WING needs to be rotated. But the Planning module has to figure out *how* to rotate the wing.

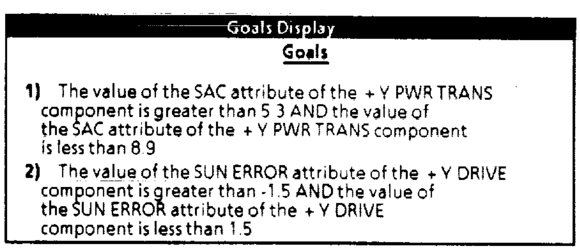

Figure 7. The Goals display.

The Planning module takes the top ranked goals and tries to find a course of action or actions that would satisfy the goals. The Planning module searches the knowledge base for events that indicate they would satisfy the goals. This is found by simulating the events looking for a trend that indicates a convergence to satisfying the goals. Once an event, or a series of events are found which could satisfy the goals, the Planning module determines what commands sent to the satellite would cause these events to occur.

Throughout the knowledge base, commanding information is "embedded" in the transition conditions for various components. The embedding of commands in transition conditions enabled the Planning module to locate commands which can potentially change the anomalous behavior of the satellite back to normal.

Once a commanding plan is found, but prior to sending any command to the satellite, the plan is verified using the knowledge base behavior specifications to validate that the anomalous conditions will be improved. Their effects are verified internally using the knowledge base specifications t confirm their effect on the satellite prior to their actual use in commanding. The Planning module is then able to determine whether to try another approach or to verify that the present planned approach is achieving the intended goals.

Once a commanding plan is verified via the knowledge base, commands are sent to the satellite (in StarPlan they are sent to the simulation knowledge base) to gather more information about the anomaly by monitoring its subsequent behavior. This process is designed to "safe" the vehicle while testing the expert system's current operating hypothesis concerning the resolution of the anomaly.

The first command found is to put the +Y WING into track mode using power system A. This command is sent, and the StarPlan monitors the telemetry data for a response. After waiting for a short while, StarPlan realizes that the track command is not working. The next command is to manually rotate the +Y WING. Once again, after waiting a short while StarPlan realizes that this command is also not working. The third command to try is the track command but with power system B. StarPlan notices that power system B is not currently on, so the command to turn it on is sent. Once power system B is on, the track command is sent. This command works (the wing position starts increasing) and StarPlan monitors the telemetry data until all of the goals are satisfied and the telemetry values return to normal (Figure 8).

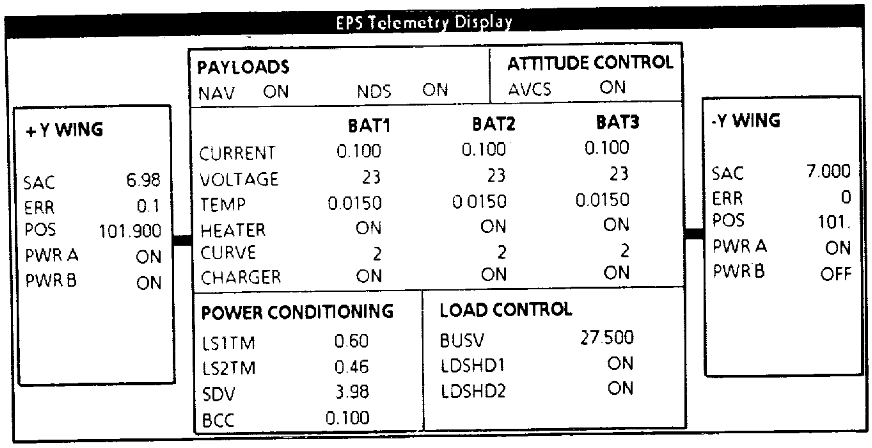

Figure 8. Normal EPS telemetry data.

CONCLUSION

Paragon is an easy to use system to build accurate functional models of a domain, such as satellites, combined with a collection of reasoning modules that use the model to resolve anomalies. Most importantly, the anomalies resolved can be completely unanticipated by human experts. The model built can be at any level of complexity, however, the more detailed the models, the finer the resolution of anomalies.

The reasoning modules described here are still being developed. As new issues arise and more complicated anomalies are tested, further enhancements or corrections become necessary. We feel confident that our reasoning approach will be able to handle many difficult to solve anomalies.

StarPlan is a prototype expert system that can handle faults on board a satellite, with only the Electrical Power Subsystem currently being modeled. Numerous anomalies have been tested with StarPlan, all of which have been resolved correctly. Further extensions to StarPlan are expected, with a complete functional model of a satellite being our ultimate goal.

REFERENCES

1. Blasdel, Arthur, "Automated Fault Handling of a Satellite Electrical Power Subsystem Using a Model-Based Expert System", Ford Aerospace Corporation internal report, Sunnyvale, California, 1988.

2. Ford Aerospace Corporation internal report, "The Paragon Development System User's Guide", Sunnyvale, California, November, 1988.

3. Hendler, James A., *Integrating Marker-Passing and Problem Solving*, Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 1988.

4. Sathi, Arvind; Fox, Mark; Greenberg Michael, "Representation of Activity Knowledge for Project Management", IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, Vol. PAMI-7, No. 5, September, 1985, pp. 531-552.

5. Wilensky, Robert, *Planning and Understanding: A Computational Approach to Human Understanding*, Addison-Wesley, Reading, MA., 1983.