C. Kitts. "Managing space system anomalies using first principles reasoning." IEEE Robotics & Automation Magazine*, Sp. Issue on Automation Science, v 13 no 4, December 2006.*

Managing Space System Anomalies Using First Principles Reasoning

Detecting, Diagnosing and Resolving Problems that Occur in Satellites and their Ground Networks

Dr. Christopher Kitts

Address for Correspondence – Robotic Systems Laboratory, Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053; Phone: 408.554.4382; Fax: 408.554.5474 (use cover sheet); E-mail: ckitts@scu.edu

Key Words – Anomaly Management, Fault Detection and Diagnosis, Health Analysis

Anomalies are unexpected conditions that occur in a functional engineering system. They must be detected, diagnosed and resolved in order to maintain the system in its functional role. Managing anomalies in space systems is particularly challenging given their complexity and their remote orbital environment. In this article, we describe our recent work in applying first principles reasoning approaches for the broad types of anomalies often encountered in complex space systems; specifically, we review extensions made to existing model-based fault detection and diagnosis techniques in order to accommodate problems other than faults and in order to resolve encountered anomalies. We also describe the software algorithms we've developed to implement this approach. Finally, we review the experimental results of applying this reasoning system while conducting mission operations for the Sapphire spacecraft, which was launched in 2001; during these experiments, the entire space system was modeled and analyzed to include both the satellite and the distributed ground operations network.

Introduction - Space systems are instrumental in generating, processing, and delivering a wide variety of products and services. A spacecraft's global view, its location above the atmosphere, and its distinctive environment all provide unique characteristics that can be exploited for commercial, civil, and military applications. Satellites are routinely used to monitor the weather, to provide communications services, to broadcast navigation signals, and to explore the solar system.

Unfortunately, the very attributes that make space systems attractive also pose considerable challenges to their efficient operation. In particular, ensuring the health of a system and managing anomalous conditions is exacerbated by the extreme nature of the space environment and the need to remotely operate the system without the luxury of direct inspection. Exacerbating these issues is the complexity of modern spacecraft and their distributed ground support networks, limitations on sensor information and configuration control, intermittent system connectivity due to orbital motion, and limitations on on-board resources such as power, communications bandwidth and computational capability.

Historically, system anomalies have been managed through the use of human-based "experiential" reasoning techniques. Highly trained and experienced engineers embed their compartmentalized understanding, rules of thumb, intuitions, heuristics, and past experiences into a loose knowledge base composed of procedures, diagrams, handbooks, manuals, and remembered information. Widespread reports in the space operations literature, as well as years of the author's own experience in operating a number of space systems, attest to the significant drawbacks of this approach. Human-based experiential systems suffer from high training and staffing costs, sensitivity to personnel changes, the impacts of human error, the inability to reuse knowledge and procedures across lifecycle phases and missions, the sensitivity of the knowledge base to small changes in the system, and many other factors [1-3]. Together, these drawbacks can result in on-orbit operations costs that constitute 25-60% of overall mission lifecycle costs [4]; for the $100 billion space industry, such system operations costs range in the tens of billions of dollars annually [5]. Declining federal outlays for space projects and increased market pressures on commercial space ventures are forcing the space industry to lower these costs. As a result, new approaches for detecting, diagnosing and responding to system anomalies are of great interest.

Reasoning from First Principles - To address the drawbacks of traditional experiential reasoning approaches, significant research has been performed over the past three decades in the field of Model-Based Reasoning (MBR). In MBR, reasoning conjectures are computed from fundamental design information regarding the design of the engineering system; as such, MBR is often defined as reasoning from first principles. For example, a *system description* is used to define the behavior of each component within a system and to declare the connectivity (e.g., the structure) of the components. With this information, the performance of the system can be modeled or simulated, thus allowing the predictions of output values given values for the system's inputs and an assumption that the system is operating nominally.

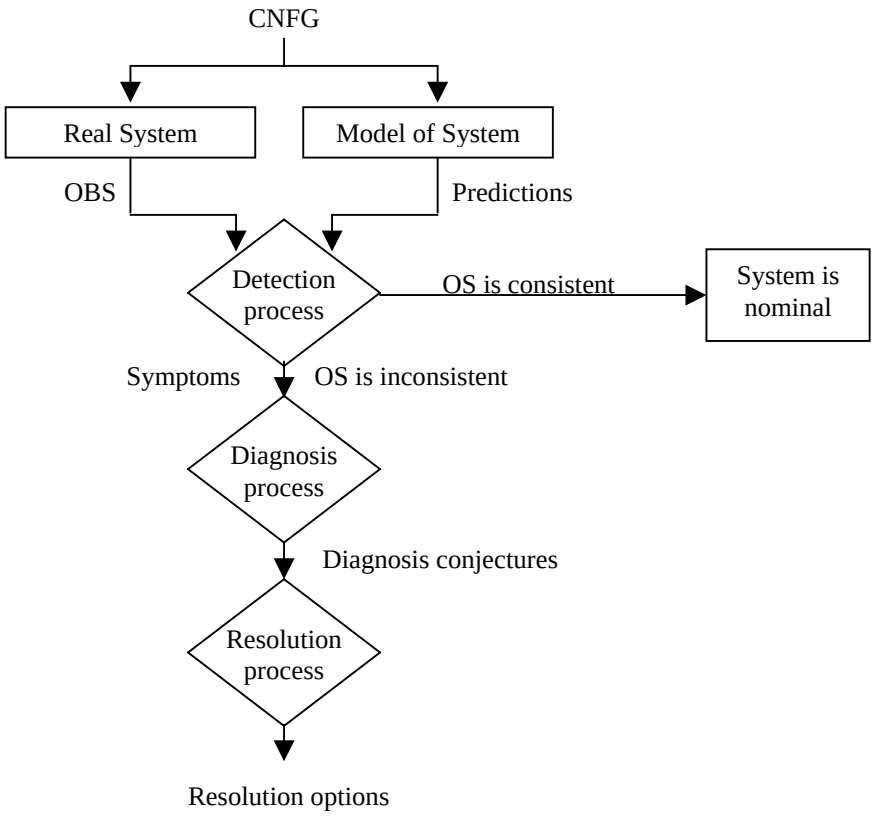
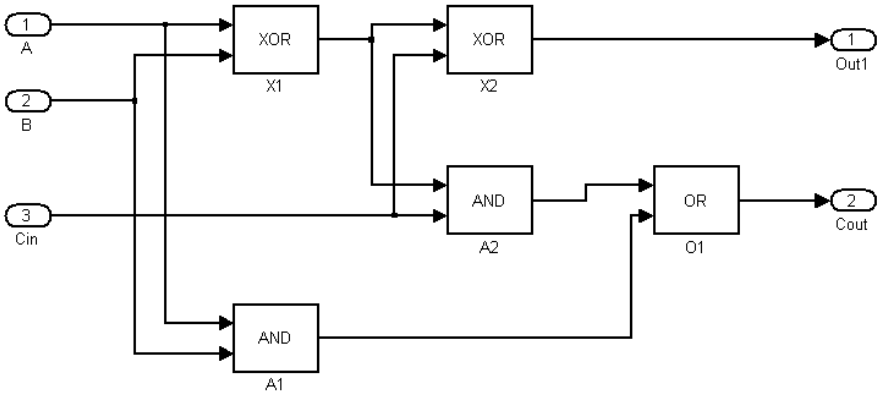


Figure 1 – Anomaly Management Process Flow. Anomaly detection is a continuous process through which the observed and predicted values of system signals are checked for consistency within the operational system (OS; the combined set of design-time and operational-time assumptions about the system). If consistent, the system is considered to be nominal. If inconsistent, a diagnosis process is initiated in order to determine anomalies whose existence can explain the observed inconsistencies. These diagnosis conjectures are provided to a resolution process, which computes viable resolution options such as reconfiguring the system or relaxing the mission requirements.

One of the most significant contributions in the field of MBR is a formal theory of fault detection and diagnosis [6,7]. This theory defines a *fault* as a condition within a component that prevents it from performing in accordance with its explicitly defined (e.g., modeled) behavior. Stated formally, the definition of a behavior prescribes a specific value of a component's output signal given the values of its inputs; mathematically, the behavior is a constraint on the output value. *Detection* of a fault using an MBR technique is accomplished by comparing the outputs of the real system to the outputs of the modeled system; as shown in Figure 1, an inconsistency between these values is interpreted as the *symptom* of a fault (given a number of assumptions, such as the accuracy of the model, etc.). A crucial distinction between MBR and other approaches to reasoning about faults is that MBR

exploits models of proper functionality (which are developed extensively in the design phase of a system) rather than attempting to enumerate all possible failure modes and deduce the resulting symptoms in order to drive the detection process [8-10].

FULL ADDER CIRCUIT



This system is a conventional full adder circuit, and it has been used as a prototypical system for demonstrating MBR fault detection and diagnosis. Consider the case in which the input signals are: A=1, B=0, Cin=0. For nominal operation, a standard model of the system predicts that the output of the system should be: S_{model}=1, $Cout_{model}$=0. However, let's assume that observations of the system show that S_{obs}=0 and $Cout_{obs}$=0. The fact that the expected and observed values of S are inconsistent establishes S_{obs} as a symptom. The interpretation of this and the subsequent reasoning are shown below for the established theory of fault detection and diagnosis and for our new theory of anomaly management.

a. Conventional Fault Detection and Diagnosis Theory – Conventional theory interprets the system as the symptom of a component fault (e.g., one of the components in the system is not behaving as defined in the model). The diagnosis process identifies components whose relaxed behavior constraints lead to the observed outputs as a possible result. For this case, this process leads to the following possible single-component diagnoses: {X1, X2}.

b. Extended Theory of Anomaly Management – Our extended theory allows us to consider hazardous component operating conditions and misconfigured inputs as possible causes of a symptom; for this example, let us assume that the intended temperature range for each component is -40°C to 70°C given that this is the range for which their behaviors are defined. Our theory therefore interprets the symptom as the indication that an anomaly exists. The subsequent diagnosis process identifies assumptions whose relaxation leads to the observed outputs as a possible result; these assumptions may be regarding the behavior of the components, the operating conditions of the components, and/or the values of the input configuration. For this case, this process leads to the following possible single-component diagnoses: {X1 faulted, X1 temperature hazard, X2 faulted, X2 temperature hazard, input C misconfigured}. Furthermore, the theory defines the conditions for appropriate resolutions, such as reconfiguring input C or sacrificing the full adder functionality; if the system included redundant components and/or heaters, reconfigurations taking advantage of these resources would be methodically generated as options.

Note – It is worth noting that MBR approaches are more precise than the experiential diagnostic heuristic "everything upstream of the symptom." Such a heuristic would also imply that A or B could also be misconfigured, but this is not possible given that these diagnoses would lead to a value of $Cout_{model}$ that is not consistent with the observed value. Furthermore, it is worth noting that the diagnostic heuristic "everything upstream of the symptom but not upstream of any good output values" is simply wrong since in this case it would exclude X1 as a valid diagnosis.

Figure 2 – Model-Based Reasoning Example. This example shows the results of applying MBR techniques to a simple system. Results are shown for both the conventional theory of fault detection and diagnosis as well as for our extended theory of anomaly management.

Once the symptom of a fault has been detected, a *diagnosis* process isolates the specific components that may be faulted. The MBR approach to identifying these components typically relies on a process known as *constraint*

relaxation. For a given component suspected of being faulty, its behavioral constraint is relaxed, which means that its output is permitted to range over its set of all valid output values. The system is then re-simulated in order to compute new values for the system's outputs. Depending on the nature of the system, the range of valid values from the possibly faulted component may lead to a range of possible values for one or more of the system's outputs. The observed values from the real system are compared to these new simulated values/ranges; if the observed values are members of the simulated sets of possible values, then the assumed component fault is, in fact, a valid diagnosis. Figure 2a provides an example of fault detection and diagnosis for a very simple digital logic circuit.

MBR has been applied to fault detection and diagnosis in a variety of fields ranging from electronic circuitry [11,12] to spacecraft health management [13-16]. These applications have motivated numerous extensions to the basic theory in order to incorporate empirical knowledge about failures [11,17,18], to generate optimal sensing plans for efficient diagnosis [12], to address computational loading through the use of hierarchical models and truth maintenance [19-21], to incorporate time-varying behavior and observations into the analysis [22], and to address the task of fault recovery [23-24].

Motivation for an Extended MBR Theory – Our work involving the operation of a variety of spacecraft and other complex robotic systems has motivated several extensions to the aforementioned theory of fault detection and diagnosis. One specific example of this involves our desire to use MBR to establish a consistent conceptual framework for reasoning about the variety of problems that are routinely encountered when operating complex satellites through networks of geographically distributed communication stations. To wit, *many things go wrong, and not all of them are faults* given the formal definition of a fault in the previously discussed theory. These other problems must also be detected, diagnosed, and ultimately resolved in order to maintain the system's ability to perform its mission.

What about Component Specifications? Consider the role that specification limits play in defining a component's behavior. For example, a component's defined behavior is typically defined assuming specific ranges for temperature, input voltage, etc. A component often has ranges over which the behavior is completely undefined. For example, an AND gate may have no defined behavior if its temperature specifications are exceeded. Furthermore, a component may have different behavioral definitions for different ranges; for example, a simple model of a transistor might have two behavioral definitions, one as an amplifier and the other as a switch, depending on whether or not the component is saturated.

Overall, we see that, distinct from physical possibility, the designer may *intend* that the component operate in a subset of the ranges for which behavioral definitions exist. *If the component is not operating as intended, this is a problem, and it is not a fault since this is not a case of a behavioral malfunction.* For example, consider an AND gate with a maximum temperature specification and intended temperature limit of 70°C. If the temperature is 75 °C and the gate's output is HIGH for all input combinations, then the behavioral constraint is not violated since it is only defined for temperatures under 70°C. However, it is clearly an unintended functionality, and its cause is due to the fact it is being operated outside of its temperature limits. Furthermore, even if the output appeared to behave with AND functionality, there is clearly still a problem since there is no guarantee that this may continue. Therefore, we see that incorporating *intention* into the model of the system is a crucial extension that enables a model-based analysis of these common situations. We have introduced the term *hazard* to formally indicate that a specific component attribute is operating outside of its intended operating range.

This understanding of how component operating conditions dictate the applicability of behavioral models leads us to incorporate several new features into our MBR theory. First, our model of the system must be able to represent behavioral conditions as well as the designers' intended constraints on these conditions. Second, we must be able to estimate these component conditions, either through observation or simulation, and then signify if these attributes are in unintended states (e.g., hazards). Third, our modeling environment must be able to select and simulate the appropriate behavioral constraint, if any, given the estimated value of each component condition.

What about Improper Configurations? Another problem that arises involves equipment that is not properly configured. While it may be easy to dismiss these problems as operator errors that are not worthy of formal consideration, their all too frequent (and often embarrassing) occurrence and our need to efficiently detect, diagnose and resolve them have motivated us to extend our conceptual framework to address these problems.

To do this, we explicitly represent a system's configuration in the description of the system. This configuration is an assumption regarding the system's inputs, and these assumptions may or may not be true (just as we have assumptions regarding component behavior and the intended operating ranges of the components). In a space system consisting of a complex spacecraft (with hundreds or thousands of inputs) and a geographically distributed

ground support network with several complicated communication stations (with tens to hundreds of inputs), it is easy to realize that the assumed configuration may not be explicitly verified at all times, especially given that many of these inputs may not be directly observable via automated data collection. We have introduced the term *misconfiguration* to formally indicate that a specific system configuration has a value other than what has been assumed.

Given the explicit representation of our configuration, we expand our MBR reasoning system in several additional ways. Most importantly, we must be able to efficiently identify which configuration inputs affect a symptomatic output, and we must be able to re-simulate the system over the range of all possible values for those inputs to see if they explain the symptom. If they do, then the fact that such a configuration input is misconfigured becomes a possible diagnosis.

Three Types of Anomalies - All three of these types of problems occur regularly in the field operation of complex engineering systems such as space systems. We formally term these problems *anomalies* with the three types of anomalies being faults, hazards, and misconfigurations. These anomalies are distinct; they may exist independently of each other, and they may interact through causal relationships. All are potential threats to the health and performance of the system, and each has differing implications regarding their effect and possible remedies. Being able to explicitly and formally detect, diagnose, and resolve each is fundamental to effectively controlling space systems in an efficient manner.

An Extended Theory of Anomaly Management – We have formalized our expanded set of operational anomalies, as well as a complementary suite of resolution actions, into a new, more comprehensive, model-based theory of anomaly management [25-26]. Like the previous theory of fault detection and diagnosis, this conceptual foundation uses a consistency-based approach that identifies and resolves inconsistencies among assumptions in the model of the system and observations of the real system. Key elements of the theory include:

- The *engineering system description*, a collection of design-time model information indicating the systems structure, behavior, and intended use.
- The *operational system description*, a collection of operational-time model information that includes the engineering system, the intended application requirements, and real-time configuration and observation data.
- The definition of *anomaly predicates* for faults, hazards, and misconfigurations.
- The definition of *resolution predicates* for formally over-riding operating constraints and altering a mission application.
- Formal definitions for a detected *symptom*, a *diagnosis* conjecture, and a *resolution* action.

Although space limitations prevent a complete discussion of the theory in this article, Figure 3 explains the formal definition of a symptom as represented using first order logic; this definition is applied during the anomaly detection process. The detection process involves identifying any inconsistency between observations of the real system and the model of our system, to include a) the system's structure, behavior and intended operation, b) the intended application of the system, c) our belief that no anomalies exist, d) our initial policy of not over-riding any operating constraints or relaxing any mission requirements, and e) our understanding of the current configuration.

Diagnosis is also a reasoning process grounded in logical consistency. In effect, diagnosis implies finding all possible anomaly conjectures that relax an assumption in our model in order to re-establish consistency between the model and our observations. For example, assume that a fault or hazard for a particular component relaxes the component's output value; if this effect ripples through the system, as determined via simulation, in order to support the real observations as one possible effect, then the fault and hazard are both viable diagnoses.

The generation of resolution options continues the use of logical consistency in order to determine the options that re-establish consistency between the model, the observations, and the belief in a specific diagnosis. Resolution options include combinations of system reconfigurations (e.g., swap out a faulted component with its redundant unit, turn on the heater for an under-temperature component, etc.), constraint over-rides (e.g., explicitly accepting a violated component operating condition), and mission alterations (e.g., explicitly relaxing one or more system requirements).

Figure 2b provides an example of these reasoning criteria for a very simple system. It is interesting to note that resolution is a function of mission requirements whereas diagnosis is not; that is because resolution is a control action that allows the mission to be accomplished, while diagnosis is an estimation process that attempts to characterize the system's state independent of its use. It is also worth noting that the theory does not rely on pre-defined redundancies or roles (like a component-specific heater) since a fundamental behavioral analysis is performed during resolution; so, functional redundancy is naturally supported such that a low temperature condition

might be satisfied by turning on the component heater or perhaps by turning on neighboring components that might also serve to heat up the component being considered (we note, however, that pre-defined redundancies and roles are certainly of value in focusing the search in a large problem space).

Definition. For an operational system, a *symptom* is a member of OBS or ATT such that:

$$\begin{aligned} &\mathbf{CNFG} \cup \{\neg \mathbf{MIS(a)} \mid \mathbf{a} \in \mathbf{ATT}\} \\ &\cup \mathbf{SD} \cup \{\neg \mathbf{AB(b)} \mid \mathbf{b} \in \mathbf{BEH}\} \\ &\cup \mathbf{CD} \cup \{\neg \mathbf{HAZ(c)} \mid \mathbf{c} \in \mathbf{CONSTR}\} \cup \{\neg \mathbf{OVR(c)} \mid \mathbf{c} \in \mathbf{CONSTR}\} \\ &\cup \mathbf{RD} \cup \{\neg \mathbf{ALT(r)} \mid \mathbf{r} \in \mathbf{REQ}\} \\ &\cup \mathbf{OBS} \end{aligned}$$

is inconsistent.

Interpretation**.** A symptom is an observed or predicted state value (e.g., OBS [observation] or ATT [component attribute]) that is inconsistent with the statements in the operational system definition. This definition is the union of the following logical statements (e.g., constraints):

- $\text{CNFG} \cup \{\neg \text{MIS(a)} \mid \text{a} \in \text{ATT}\}$: The list of configuration inputs and their assumed values, with the assumption that none of these inputs are misconfigured.
- $\text{SD} \cup \{\neg \text{AB(b)} \mid \text{b} \in \text{BEH}\}$: The list of structure and behavior statements with the assumption that none of the behaviors are abnormal (e.g., faulted).
- $\text{CD} \cup \{\neg \text{HAZ(c)} \mid \text{c} \in \text{CONSTR}\} \cup \{\neg \text{OVR(c)} \mid \text{c} \in \text{CONSTR}\}$: The list of intended constraints on component operating conditions, with the assumption that none of these constraints have been violated (e.g., are hazards) or have been intentionally overridden.
- $\text{RD} \cup \{\neg \text{ALT(r)} \mid \text{r} \in \text{REQ}\}$: The list of intended mission requirements, with the assumption that none of these requirements have been relaxed (e.g., altered).
- OBS: The list of observed values.

Figure 3 – Formal Definition of a Symptom. The MBR Anomaly Management Theory consists of a number of formal definitions of an engineering system, an operational system, anomaly types, resolution actions, and management tasks. The first management task is detection, which involves the identification of a symptom and which is expressed here using first order logic. As can be seen, the definition relies on analyzing the consistency among our observations of the real system and our modeled assumptions regarding the design and use of the system.

Overall, we emphasize the fact that *the distinction among anomaly types is critical* since the cause of, the defining criteria for, and the solution to different types of anomalies are distinct. *Nevertheless, we are able to unify their treatment in a single theory* given that each type of anomaly represents the violation of an assumption in the system model. Therefore, our anomaly management algorithms seek to identify inconsistencies among assumptions (detection), to isolate specific assumptions within the system model that cause these inconsistencies (diagnosis), and to propose actions or select assumptions to relax in order to re-establish consistency within our model (resolution).

Implementing the Theory – Our current algorithmic implementation of this theory has been developed as a toolbox for Matlab given our extensive use of this commercially available programming environment for multi-physics system modeling and analysis and our desire to use it for model composition and high-performance simulation tasks relating to this work. To properly implement the theory, this implementation requires a representation of the system as described by the theory's engineering and operational system descriptions, and it allows mathematical statements to be relaxed using the anomaly and resolution predicates. Given this, the algorithms methodically compute symptoms, diagnoses and resolutions in accordance with the criteria specified in the theory's formal definitions [26].

Regarding the representation of the system model, a composable modeling approach has been adopted by using the structure data type within Matlab in order to define components with their inputs, outputs, internal states, behavioral definitions, etc; simple statements equating the values of component outputs to downstream component inputs are used to represent connections.

Anomaly Management Algorithms – The anomaly management algorithms are modularly implemented, with separate algorithms for detection, diagnosis, and resolution.

The anomaly detection process is executed periodically, typically after a standard set of telemetry has been received during routine monitoring and certainly after a command has been sent.[1] For a given input configuration, the model is evaluated in order to predict system outputs. These outputs are then compared to telemetry from the actual space system. Specific checks include whether the observations directly conflict with any of the configuration assumptions, whether the observations violate any of the intended component operating conditions, or whether the observations are inconsistent with the predicted observation outputs.

The diagnosis algorithm is invoked only when the detection algorithm identifies a symptom. As shown in Figure 4, the algorithm is implemented as a two-stage process given the appropriateness of two distinct decision-making approaches:

- In the first stage, a production rule process compares observations with assumed configuration values and intended component operating ranges. This is an efficient computational process that involves direct inspection of these stored assumptions. Inconsistencies lead directly to misconfiguration and hazard diagnoses, respectively.
- The second phase of the anomaly diagnosis algorithm addresses symptoms generated by inconsistencies between the predictions and observations of the system state. These symptoms demand the application of a constraint relaxation process in order to identify their associated diagnosis conjectures. This process is implemented by systematically relaxing behavioral, operating constraint and configuration assumptions within the OS and re-evaluating the system until the predicted state of the system is once again consistent with observations. To date, versions of this algorithm phase have been implemented in order to identify sets of diagnoses for multi-symptom single-remaining-anomaly cases.

The resolution process only executes when the diagnosis algorithm returns a set of diagnosis conjectures. For a specific diagnosis, the algorithm is performed by systematically considering mission alterations, constraint overrides, and new configurations. The first two are easily generated: if any mission-critical system outputs or intended component conditions are violated, simply accepting these facts by relaxing the mission or overriding the constraints become resolution options. Of course, these are usually accepted only as a last resort. Therefore, it is critical to evaluate reconfiguration options. This is done by re-evaluating the model for permutations of configuration options upstream of the diagnosis; new configuration combinations that result in the satisfaction of a violated mission requirement are saved as possible resolution options. Finally, the three types of resolution options are considered in systematic combinations; minimal combinations that re-establish consistency within the system (given the assumed diagnosis) are saved as valid resolutions.

Performance of the Algorithms – Speed of processing is of obvious importance given the need to quickly return a space system to a healthy state in the event of an anomaly. Executing on a Pentium IV PC, the current implementation of the algorithms execute detection, diagnosis and resolution (for a specific diagnosis) on the order of seconds to tens of minutes. Broken down, detection generally occurs on the order of hundredths of seconds, diagnosis typically takes seconds, and resolution ranges from seconds to tens of minutes. This is for systems with 20-30 components (which is the general resolution of operational anomaly management performed by humans) which typically require hundreds of mathematical constraints.

To put this performance level into the proper perspective, it is important to note two facts. First, state-of-the-practice anomaly diagnosis and resolution in the space industry often takes hours, days, or even weeks given the reliance on manual and experiential processing and the fact that the engineers that perform these tasks are generally not the operators that perform realtime command and telemetry operations. Second, our current implementation can be significantly improved given that computational performance has been sacrificed in order to promote exploratory implementation, to examine alternate modeling and computational approaches, and to assess the computational similarities in computing diagnoses and resolutions for different classes of anomalies. Furthermore, the current software runs in an interpreted mode as Matlab scripts on a multi-tasking computer rather than as a set of compiled executable functions on a dedicated workstation; optimization of the algorithms is expected to improve their computational speed by more than a factor of 10 [27]. Given these facts, we believe that MBR anomaly management is capable of providing valuable decision support in current space system operations environments.

[1] For satellite configuration control applications, cycle times for comprehensive health analysis is typically on the order of 1-100's of minutes; this is often driven by limited availability and bandwidth of the communications link.

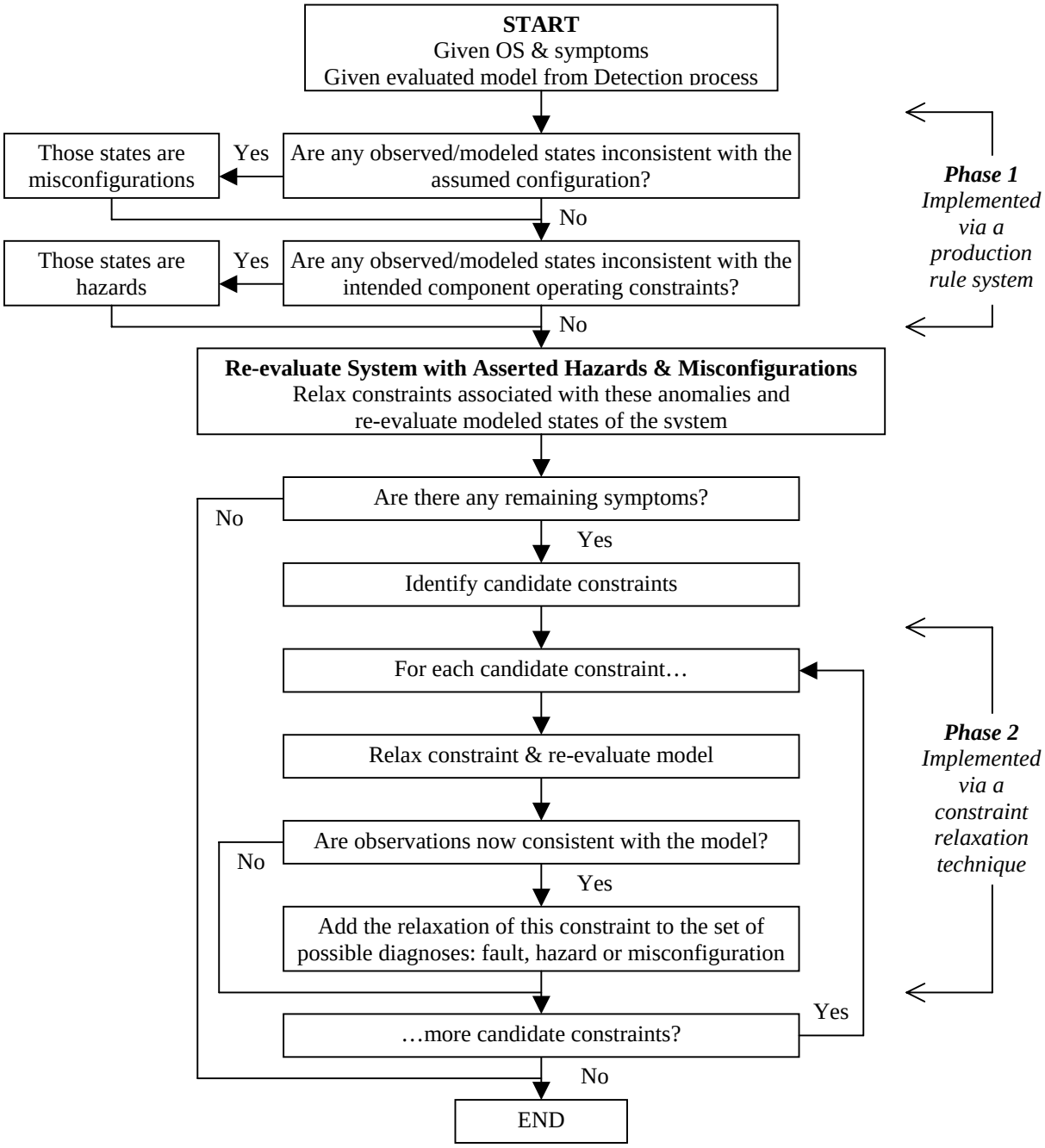
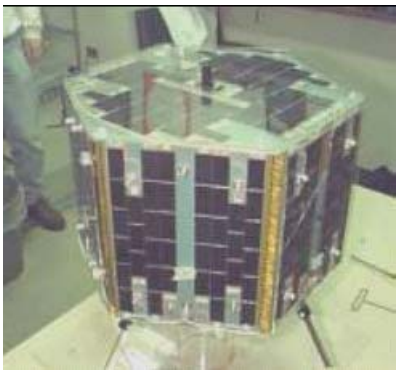


Figure 4 – Overview of the Diagnosis Algorithm. This flowchart depicts the general flow of the two-phase diagnosis algorithm. In the first phase, a production rule computational process compares observed and modeled states with configuration and operating condition constraints. Given any identified anomalies from this process, any symptoms that remain unexplained are diagnosed in phase 2 using a constraint relaxation computational process. The result of the diagnosis algorithm is a set of possible diagnoses.

Commentary Regarding Anomaly Management Processing – There are several subtle characteristics of the anomaly management process worth highlighting. First, different types of anomalies can lead to the same symptom. For example, consider an AND gate that has an incorrect HIGH output given to LOW inputs. The gate may be faulted in a 'stuck-HIGH' condition, or it may have an out-of-limit temperature such that its behavior is no longer guaranteed to act like the AND function, or perhaps the inputs are misconfigured and they are both really HIGH. This many-to-one correlation makes operational anomaly management a difficult task. Second, the ability of an anomaly to be observed and to affect the rest of the system is a function of the system's configuration. Finally, because anomaly observability is configuration dependent, operational anomaly management is necessarily parsimonious; if there is no indication of an anomaly, then no anomaly is assumed given that mission critical systems cannot be arbitrarily reconfigured in order to verify a healthy state.

MBR techniques are compatible with these characteristics, and in many cases it is these factors that highlight the power of MBR. Through its ability to systematically evaluate a system's operation from the first principles codified in a design description, MBR has the potential to be far more precise in its diagnosis and resolution conjectures compared to an experiential system, which often diagnoses problems using an "anything upstream" strategy.

Applying the Theory to the Sapphire Space System – We have experimentally verified and validated our anomaly management reasoning system by applying it to the operational control of the Sapphire space system, to include the satellite itself as well as its distributed ground network. Launched in 2001, Sapphire was developed by students at Stanford University and supports a variety of missions to include digital communications, Earth photography, and sensor characterization [28]. As depicted in Figure 5, the comprehensive Sapphire space system consists of the Sapphire microsatellite, a standalone wireless test device (which served as an always available satellite surrogate), several automated communication stations located throughout the United States, and a centralized mission control complex in the Space Technology Center building in the NASA Ames Research Park, Moffett Field, California. The communication stations are remotely controlled via the internet by operators in the mission operations center, and amateur radio is used to communicate between these stations and the satellite when it is locally in view. The ground segment portion of this system, developed by students at Santa Clara University, is used to support the operation of a variety of other spacecraft and robotic missions [29,30].



(a) The Sapphire Microsatellite

(b) Mission Operations Center

Figure 5 – The Sapphire Space System. (a) The Sapphire microsatellite was launched in 2001. It has been operated by hundreds of students and amateur radio operators around the world since that time. Several geographically distributed, internet-connected communication stations have been used to support command and telemetry operations with the satellite. (b) Student operators at the Center for Robotic Exploration and Space Technologies in the NASA Ames Research Park perform operations and mission management tasks for Sapphire and a variety of other spacecraft and remote robotic system.

Models for each element in this system have been composed, and the model-based anomaly management system has been used extensively for standard ground-based operation of Sapphire during the past three years; to our

knowledge, this is the first time that MBR technology has been applied to manage anomalies across a comprehensive space system that includes both a ground-based command, control, and communications segment as well as a space segment. In addition, a compiled set of model parameters were incorporated into an advanced production rule system that is installed on the satellite and which was verified and validated during more than a year of mock operational experiments performed prior to launch, with the satellite being operated as if in orbit [31, 32].

Fortunately (or unfortunately, depending on the applicable point of view), many real and unanticipated anomalies involving both the satellite and the ground segment occurred during these contacts, providing many wonderful opportunities to exercise the anomaly management system in a realistic setting. Many of these anomalies provided feedback for further refinement of the design models used by the anomaly management software in order to improve the resolution of reasoning or enhance processing speed through abstraction of unnecessary detail. Several of the most interesting anomalies are summarized in Table 1. In these and all of the following cases, the anomaly management software was able to successfully detect symptoms and generate a valid set of possible diagnoses and resolutions:

- Reset satellite due to low batteries, causing an incorrect CPU time.
- Improperly executed payload procedure in which data collection was attempted from a unit that was off.
- Unresponsive operating systems on the communication station computer.
- Misconfigured IP address on the communication station computer (the host institution changed the IP address without notifying the operations team).
- Power outage at the communication station facility.
- Out of date Keplerian elements used by the communication station autotrack software.
- Incorrect time setting for the communication station computer leading to inaccurate autotrack computations.
- Overheating of the transmitter amplifier leading to undesired performance.
- Misconfiguration of TNC settings by the operator.
- Misconfiguration of autotrack software settings by the operator.
- Failure of an antenna positioning servomotor (on several occasions).
- Improperly executed operations procedures resulting in misconfigurations.

Anomaly Examples - To provide insight into the application of the anomaly management algorithms, let us consider two specific anomalies.

The first is the on-board sensor problem detailed in the third entry of Table 1. Figure 6 shows a simplified block diagram of the model for the applicable portion of the Sapphire satellite. In this example, the desired operational task was to collect data from experimental infrared sensors that are one of the primary payloads on the Sapphire spacecraft. When data was acquired from these parallel sensors, they were inconsistent with the expected values thereby triggering an initial, automated diagnosis process that returned three possible diagnoses. Of these three, a possible resolution (short of altering the mission) existed for the "misconfigured sensor" conjecture. This resolution, to command the sensor to its enabled configuration, was executed and solved the problem. The speed of the diagnosis process, which executed in approximately 3.5 seconds, allowed the operator to properly configure the satellite, collect the desired sensor data, and complete other operational tasks within the 12-minute contact window.

For this example, it is instructive to consider the diagnoses that were not made. Diagnoses relating to component power were not made given that power was being properly supplied to other units within the satellite. It is also interesting to note that faulty sensors were not a diagnosis. This is because the initial diagnosis that was executed assumed only a single anomaly had occurred; because the sensors are in parallel, both would have had to fail in order to produce the observed data.

For our second example we consider the "no acquisition of satellite signal" problem detailed in the second entry of Table 1. In this example, no radio frequency signal was received from the satellite at the initiation of an attempted contact. Successful radio contact requires a long serial string of equipment (e.g. computers, encoders, transmitters, antennae, etc.) to be properly configured, each of which must be properly behaving (not faulted) with the appropriate configuration (not misconfigured) and operating conditions (not subjected to a hazard). For this particular situation, 63 diagnoses were returned in response to the "no acquisition" symptom. These were generated in less than 2.5 seconds and included a wide range of fault, hazard, and misconfiguration conjectures for the components, their operating states, and their configuration inputs. Rather than initiate a full resolution process for each of these, we initiated a set of communication station "loop-back tests" (such as attempting to establish radio contact with a test transmitter co-located at the communication station) in an attempt to reduce the number of

diagnosis conjectures.[2] These tests allowed us to eliminate diagnoses for all modeling assumptions except those involving the antenna pointing system. Upon manual inspection, the problem was ultimately a misaligned antenna, which had been moved by a maintenance worker (certainly an unplanned event!).

Table 1 – Examples of Managed Anomalies

Contact Operation	Anomaly Management Reasoning Products	Conclusion of Operations Team
Task Basic State of Health Check	**Symptoms** Incorrect CPU Time **Diagnoses** D_1: Hazardous Radiation Reset of CPU Time D_2: Hazardous Power Usage Reset of CPU Time D_3: Misconfigured CPU Time **Resolution** R_{3a}: Reconfigure CPU Time via time reset command	**Interpretation**: Condition occurs due to CPU reset or clock mismanagement. Conclusion was CPU reset, most likely due to power cycle (most likely resulting from aging batteries and significant communications usage by amateur radio users). Resolution was to reconfigure CPU clock for nominal operations and to suspend amateur radio community use, resulting in lower power demand. **Algorithm Performance**: Full reasoning process executed in ~2.7 seconds. Anomaly properly detected, diagnosed, and resolved.
Task Initiation of Contact	**Symptoms** No Signal Received **Diagnoses** 63 diagnoses (2.43 sec) Active testing narrowed diagnoses to antenna pointing system.	**Interpretation**: This was the most conceptually interesting ground segment anomaly. The reasoning system properly isolated the antenna pointing system as being anomalous. Ultimately, the problem was an antenna that had a significant mechanical misalignment. **Algorithm Performance**: The reasoning system identified the proper subsystem via active testing, but ultimately failed to generate the proper explanation due to the limits of the system description.
Task Collection of IR Sensor Data	**Symptoms** IR sensor data at max values **Possible Diagnoses** D_1: Faulted IR Sensor Enable line in Command Processor D_2: Faulted A/D converter data line in Telemetry Formatter D_3: Misconfigured IR Sensor Enable Configuration **Resolutions** $R_{1a, 2a, 3a}$: Relax requirement to collect IR sensor data. R_{3b}: Reconfigure the IR Sensor Enable Setting	**Interpretation**: Data from the IR sensor was invalid (e.g. inconsistent with predicted values). Diagnoses included a faulty CP IR enable line, a faulty A/D board data line, and a misconfigured IR enable setting; note that IR sensor faults were not computed since this would require 2 independent faults (which was not considered an option at the time). Resolutions included relaxing the mission requirement to collect IR data (appropriate for all D_n) or to reconfigure the enable setting (for D_3). The reconfigure option was exercised and valid IR data was collected, confirming the D_3 diagnosis. Subsequent human analysis of this episode confirmed that the symptomatic values were consistent with known disabled or faulted values. **Algorithm Performance**: Full reasoning process executed in ~ 3.5 sec. System directly contributed to successful and timely anomaly resolution during the < 12 minute contact.

[2] This type of active diagnosis (which exploits the previously discussed configuration dependency of our ability to draw conjectures regarding the system) is not always possible; in this case, it was permitted given that altering the communication station configuration did not interfere with any mission critical experiments on the satellite.

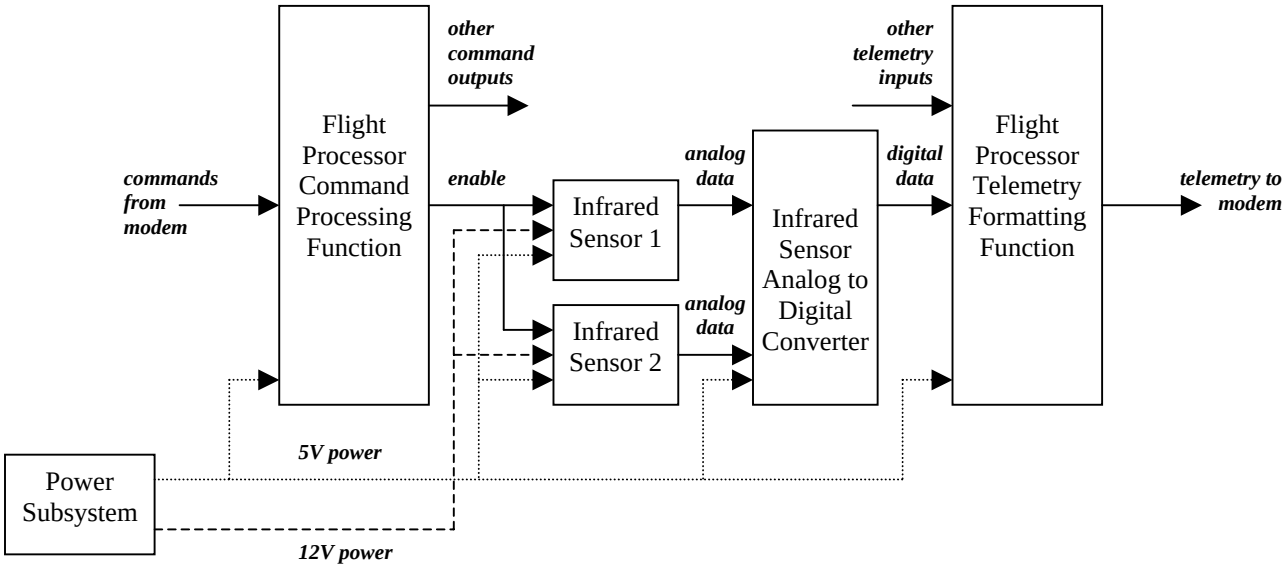


Figure 6 – The Sapphire Infrared Sensor Anomaly. This diagram shows a portion of the Sapphire spacecraft relevant to the infrared sensor anomaly. Poor data from both infrared sensors resulted in possible diagnoses that included a misconfigured enable signal from the command processor, a faulted command processor infrared sensor enable behavior, or a faulted analog to digital converter behavior. The power system was not suspected since observations showed that it was properly providing power to other components. In addition, faulted infrared sensor behaviors were not suspected since this would have required multiple faults, a situation not considered in the first pass of the diagnosis algorithm. Of the three possible diagnoses, the misconfiguration option had a resolution other than reducing the scope of the mission. This resolution was attempted through sending a command to the satellite to enable the infrared sensor line; luckily, this solved the problem.

Lessons Learned - The experiments yielded a number of "lessons learned" regarding the value of the model-based anomaly management strategy:

- The model-based approach provided a systematic approach to evaluating the complete solution space for each anomaly management task, and served as a strong complement to the advantages and disadvantages of the experiential techniques routinely used by human operators. In addition, it demonstrated the power of using symbolic representations of engineering functionality in order to draw strong conclusions regarding the state of a system.
- Human interaction with the model-based reasoning system "rubbed-off" on the human operators and improved their ability to systematically evaluate anomaly scenarios.
- Discrepancies between the conclusions drawn by human operators using experiential techniques and those by the automated model-based system often took hours to resolve (e.g., for the human operators to understand why their conclusions were incorrect and/or incomplete). It is clear that the future acceptance of model-based reasoning systems (and, in fact, any high performance reasoning system) will require tools and mechanisms for providing insight into how the results were generated.
- Development of the design models themselves was also tedious. This points to a need to develop design synthesis, editing and visualization tools. We hope to capitalize on our current use of Matlab by using it and its associated visual model-building tools for composing system design models.
- The model-based approach is not a panacea for the anomaly management process. Rather, its power was demonstrated in its ability to complement other approaches which draw from experience and employ time-saving heuristics. In addition, there will always be a tension between the need to use simple and abstract models to promote speedy computation and the need to use detailed and complex models to capture detailed nuances of the engineering system.

Future Work – We have significant plans to evolve this work in the future. Future theoretical extensions include addressing new anomaly classes, such as undesirable threats that a system may cause to external entities. We also hope to apply the theory to the design process in order to support studies of sensor placement, command resolution, etc. Improvements to our implemented processing system will focus on the incorporation of more sophisticated representation schemes such as hierarchical modeling, computational efficiency through optimized and compiled algorithms, and support of enhanced dynamic simulation and modeling frameworks using state-of-the-art tools. Furthermore, we have already initiated work to improve human use of our model-based system through the incorporation of standard model composition tools and the development of feedback that provides transparency to the reasoning conjectures.

Critical to these improvements is the application of our system to new and diverse systems that will drive our innovations. In the space system domain, we are already applying extended versions of our system to new flight projects such as the NASA GeneSat-1 spacecraft, the University of Texas at Austin's two-satellite FASTRAC formation flying mission, and Santa Clara's own ONYX autonomy demonstration microsatellite which will support on-board anomaly injection for controlled, double-blind anomaly management technology validation and which is being developed through a grant from the Air Force Office of Scientific Research. Furthermore, we are also applying our system to non-space systems such as the control of a novel, fault-tolerant vectored thruster for autonomous underwater robots.

Summary and Conclusions – In this article, we have discussed the development of a theory of anomaly management that uses fundamental models of a systems structure, behavior, and intended use in order support the detection of symptoms, the computation of possible diagnoses, and the generation of resolution actions. We have implemented a software-based reasoning system based on this theory and have applied it to the configuration control of a space system consisting of an on-orbit satellite and a geographically distributed network of ground communication and control stations. We have found that use of this system promotes a formal and systematic analysis of possibilities when managing configuration anomalies in the Sapphire space system. Furthermore, we believe that this approach can be extended to more complex systems through future optimization of the processing implementation and through incorporation of other innovations such as dynamic and hierarchical modeling. Overall, we are convinced of the power of MBR for its methodical, physics-based examination of the problem space, for its potential to efficiently leverage design-time analytic models for operational decision tools, and for its ability to complement other reasoning approaches in a hybrid architecture for anomaly management.

Acknowledgements – This work has benefited greatly through the feedback of numerous colleagues; we are particularly indebted to Dr. Mike Swartwout for his invaluable comments and critical review regarding several iterations of this work. The reviewers of this article also contributed to its value through their constructive comments. By leveraging university-based programs to support the experimental element of this work, we have also relied on the contributions of many students in the development of the test satellite and ground systems; we are indebted to these wonderful individuals, which number more than one hundred from Santa Clara and Stanford Universities. Finally, this work has been sponsored through a variety of sources to include the Department of Defense, NASA Ames Research Center, and the National Science Foundation via Grant No. EIA0079815; any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the Department of Defense, NASA or the National Science Foundation.

References

[1] S. Hovanessian, S. Raghavan, and D. Taggart, "LIFELINE: A Concept for Automated Satellite Supervision," The Aerospace Corporation Report No. TOR-93(3516)-1, El Segundo, CA, 1993.
[2] G. Brown, D. Bernard, and R. Rasmussen, "Attitude and Articulation Control for the Cassini Spacecraft: A Fault Tolerance Overview," in *Proceedings of the 14th AIAA/IEEE Digital Avionics Systems Conference*, Cambridge, MA, November, 1995.
[3] C. Kitts and M. Swartwout, "Experimental Initiatives in Space Systems Operations," in *Proceedings of the Annual Satellite Command, Control and Network Management Conference*, Reston, Virginia, 1997.
[4] N. Ely and T. O'Brien, "Space Logistics and Reliability," *Space Mission Analysis and Design*, Eds. J. Wertz and W. Larson, London: Kluwer Academic Publishers, 1991, pp. 633-656.
[5] "State of the Space Industry," Space Publications, Reston, VA, 2005.
[6] R. Reiter, "Theory of Diagnosis from First Principles," Artificial Intelligence, vol 32, no. 1 pp. 57-95, 1987.

[7] J. de Kleer, A. Mackworth, and R. Reiter, "Characterizing diagnoses and systems," Artificial Intelligence, vol 56, pp. 197-222, 1992.
[8] Y. Kato, T. Shirakawa, and K. Hori, "Utilizing Fault Cases for Supporting Fault Diagnosis Tasks," Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies, KEW 2002, vol 2. pp. 1262-6, 2002.
[9] T. Bak, R. Wisniewski, and M. Blanke, "Autonomous Attitude Determination and Control System for the Orsted Satellite," IEEE Aerospace Applications Conference Proceedings, v 2, pp. 173-185, 1996.
[10] C. Wilklow, S. Doraisingam, E. Hansen, and J. Willis, "Citizen Explorer – The Application of an Innovative Mission Operations System," IEEE Aerospace Applications Conference Proceedings, v 2, pp. 157-162, 2000.
[11] M. Genesereth, "The use of design descriptions in automated diagnosis," Artificial Intelligence, vol 24, pp. 411-436, 1984.
[12] J. de Kleer and B. Williams, "Diagnosing multiple faults," Artificial Intelligence, vol 32, pp. 97-130, 1987.
[13] R. Doyle, S. Chien, U. Fayyad, and E. Wyatt, "Focused Real-time Systems Monitoring Based on Multiple Anomaly Methods," in *Proceedings of the 7th International Workshop on Qualitative Reasoning*, Seattle, WA, May, 1993.
[14] D. DeCoste, "Automated learning and monitoring of limit functions," in *Proceedings of the Fourth International Symposium on Artificial Intelligence, Robotics, and Automation for Space (i-SAIRAS-97*), Japan, July, 1997.
[15] L. Fesq, A. Stephan, and L. McNamee, "Modeling Power Systems for Diagnosis: How good is good enough," in *Proceedings of the 27th Intersociety Energy Conversion Engineering Conference*, San Diego, CA, 1992, pp 203-208.
[16] B. Williams, and P. Nayak, "A Model-based Approach to Reactive Self-Configuring Systems," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, Portland, OR, 1996, pp. 971-978.
[17] G. Friedrich, G. Gottlob, and W. Nejdl, "Physical impossibility instead of fault models," in *Proceedings of the 1990 AAAI National Conference on Artificial Intelligence, Boston, MA*, July, 1990, pp. 331-336.
[18] P. Struss, and O. Dressler, "Physical Negation: integrating fault models into the general diagnostic engine," in *Proceedings of the International Joint Conference on Artificial Intelligence*, San Mateo, CA: Morgan Kaufmann, 1989, pp. 1318-1323.
[19] R. Davis, "Diagnosis based on description of structure and function," in *Proceedings of the 1982 AAAI National Conference on Artificial Intelligence*, Pittsburg, PA, August, 1982, pp. 137-142.
[20] M. Genesereth, "Diagnosis using hierarchical design models," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, Lost Altos, CA: Morgan Kaufmann, 1982, pp. 278-283.
[21] J. Doyle, "A truth maintenance system," Artificial Intelligence, vol 12, pp. 231-272, 1979.
[22] H. Ng, "Model-based, multiple fault diagnosis of time-varying, continuous physical devices," in *Proceedings of the 6th IEEE Conference on Artificial Intelligence Applications*, Santa Barbara, CA, 1990, pp. 9-15.
[23] J. Crow, and J. Rushby, "Model-based reconfiguration: Toward an integration with diagnosis," in *Proceedings of the 1991 AAAI National Conference on Artificial Intelligence*, San Mateo, CA: Morgan Kaufmann, 1991, pp. 836 -841.
[24] G. Friedrich, G. Gottlob, and W. Nejdl, "Formalizing the Repair Process," in *Proceedings of the European Conference on Artificial Intelligence*, Vienna, August, 1992.
[25] C. Kitts, "Model-Based Space System Anomaly Management – Part 1: A First Principles Foundation," in draft for submission to IEEE Transactions on Automation Science and Engineering.
[26] C. Kitts, "Model-Based Space System Anomaly Management – Part 2: Implementation and Application," in draft for submission to IEEE Transactions on Automation Science and Engineering.
[27] E. Hall, "Maximizing Matlab Performance," University of Virginia Research Computing Support Center, Charlotteville, VA, 2003.
[28] M. Swartwout, C. Kitts, R. Twiggs, B. Smith, T. Kenny, R. Lu, K. Stattenfield, R. Batra, and F. Pranajaya, "Sapphire: A Case Study in University Class Satellites," Accepted for publication, AIAA Journal of Spacecraft and Rockets.
[29] D. Schuet and C. Kitts, "A Distributed Satellite Operations Testbed for Anomaly Management Experimentation." *Collection of Technical Papers – AIAA 3rd "Unmanned-Unlimited" Technical Conference, Workshop, and Exhibit*, Chicago, IL, September 20-23, 2004.
[30] C. Kitts, "Surf, Turf, and Above the Earth: An Aggressive Robotics Development Program for Integrative Undergraduate Education," IEEE Robotics and Automation Magazine, vol 10, no. 3, pp. 30-36, 2003.

C. Kitts. "Managing space system anomalies using first principles reasoning." IEEE Robotics & Automation Magazine*, Sp. Issue on Automation Science, v 13 no 4, December 2006.*

[31] M. Swartwout, C. Kitts, and R. Batra, "Persistence-Based Production Rules for On-Board Satellite Autonomy," IEEE Aerospace Applications Conference Proceedings, Aspen, CO, v 1, Mar. 1999, pp. 273-281.
[32] C. Kitts and M. Swartwout, "Beacon Monitoring: Reducing the Cost of Nominal Spacecraft Operations," Journal of Reducing Space Mission Cost, vol 1, no. 4, pp. 305-338, 2002.

Author Biography – Dr. Christopher Kitts is an Associate Professor at Santa Clara University where he serves as the Director of the Robotic Systems Laboratory. He is also the Director of the Silicon Valley Center for Robotic Exploration and Space Technologies, a multi-institution collaborative for robotic and aerospace research and education which is located at the NASA Research Park in Moffett Field, CA. At Santa Clara, Prof. Kitts runs an aggressive field robotics program specializing in the design, control and teleoperation of highly capable robotic system for scientific discovery, technology validation, and engineering education. These systems include underwater vehicles, clusters of land rovers, autonomous aircraft, and microspacecraft. These systems provide unique experimental opportunities for demonstrating research innovations in multi-robot systems, model-based anomaly management, and other research topics within the Lab. Prof. Kitts' professional experience includes work ranging from a research engineer to an operational satellite constellation mission controller, and he has held appointments as an officer in the U.S. Air Force Space Command, as a NASA contractor with Caelum Research Corporation, as a DoD Research Fellow at the U.S. Phillips Laboratory, and as the Graduate Student Director of Stanford University's Space Systems Development Laboratory. He holds a B.S.E. from Princeton University, an M.P.A. from the University of Colorado, and M.S. and Ph.D. degrees from Stanford University.