Intrinsic Hurdles in Applying Automated Diagnosis and Recovery to Spacecraft

James Kurien and María D. R-Moreno

Abstract*—Experience developing and deploying model-based diagnosis (MBD) and recovery and other model-based technologies on a variety of testbeds and flight experiments led us to explore why our expectations about the impact of MBD on spacecraft operations have not been matched by effective benefits in the field. By MBD, we mean the problem of observing a mechanical, software, or other system and determining what failures its internal components have suffered using a generic inference algorithm and a model of the system's components and interconnections. These techniques are very attractive, suggesting a vision of machines that repair themselves, reduced costs for all kinds of endeavors, spacecraft that continue their missions even when failing, and so on. This promise inspired a broad range of activities, including our involvement over several years in flying the Livingstone and L2 onboard MBD and recovery systems as experiments on Deep Space 1 and Earth Observer 1 spacecraft. Yet, in the end, no spacecraft project adopted the technology in operations nor flew additional flight experiments. To our knowledge, no spacecraft project has adopted any other MBD technology in operations. In this paper, we present a cost/benefit analysis for MBD using expectations and experiences with Livingstone as an example. We provide an overview of common techniques for making spacecraft robust, citing fault protection schemes from recent missions. We lay out the cost, benefit, and risk advantages associated with onboard MBD and use the examples to probe each expected advantage in turn. We suggest a method for evaluating a mission that has already been flown and providing a rough estimate of the maximum value that a perfect onboard diagnosis and recovery system would have provided. By unpacking the events that must occur in order to provide value, we also identify the factors needed to compute the *expected* value that would be provided by a real diagnosis and recovery system. We then discuss the expected value we would estimate that such a system would have had for the Mars Exploration Rover mission. This has allowed us to identify the specific assumptions that made our expectations for MBD in this domain incorrect.

***Index Terms*—Benefits, cost, expected value, fault protection (FP), model-based diagnosis (MBD), spacecraft.**

Manuscript received June 17, 2008. Date of publication July 19, 2010; date of current version August 18, 2010. This work was supported by the National Aeronautics and Space Administration (NASA), including the NASA's Intelligent Systems Program. The work of M. D. R-Moreno was supported by the Castilla-La Mancha Project PEII09-0266-6640. This paper was recommended by Guest Editor P. Struss.

J. Kurien is with the National Aeronautics and Space Administration Ames Research Center, Moffett Field, CA 94035 USA (e-mail: James.A.Kurien@nasa.gov).

M. D. R-Moreno is with the Departamento de Automática, Universidad de Alcalá, 28805 Alcalá de Henares, Spain (e-mail: mdolores@aut.uah.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2010.2052035

I. Introduction

Over the past 20 years, there has been much work in the area of model-based diagnosis (MBD). By this, we mean diagnosis systems arising from computer science or artificial intelligence approaches, where a generic software engine is combined with a model of a system's components and interconnections to generate diagnoses and, more recently, to infer system recoveries or reconfigurations. Two diagnosis and recovery systems spawned by this line of research were exercised in simulations, on hardware testbeds and during two flight experiments on the Deep Space 1 (DS-1) and Earth Observer 1 (EO-1) spacecraft. A great deal of experience was gained by developing these systems and applying them to a variety of diagnosis domains involving spacecraft, chemical processors, life-support systems, and scientific instruments. Supporting infrastructure, including modeling languages, graphical model editors, and validation tools, was created in addition to the core inference algorithms and software that performed diagnosis and recovery. In the end, however, no spacecraft project adopted the technology in operations nor flew additional flight experiments. To our knowledge, no spacecraft project has adopted any other MBD technology in operations. In short, in the ten years since the first flight experiment, the benefits we anticipated have not yet come to fruition.

This led to a series of questions. What are the costs of using MBD? What benefits are observed in practice? How can we approach the question of whether the benefits outweigh the costs for spacecraft? How are missions today approaching fault diagnosis and recovery during operations? How does the proposed MBD approach differ with the current practice, and why are not we able to leverage MBD technologies to provide greater value or lower cost? Is there something about planning software, in many ways a similar model-based technology, that has made it more successful, and what does that tell us? By attempting to answer these questions, we have identified specific assumptions that made our expectations for MBD in this domain incorrect.

This paper is organized as follows. In Section II, we present a brief description of the spacecraft fault protection (FP) problem and common practice for addressing it, followed by a brief overview of the Livingstone MBD and recovery system. We then consider Livingstone's impact on spacecraft FP practice, both in terms of the impact we expected based on its capabilities and the impact we have observed in the subsequent years. To understand the discrepancy, we express the technology's potential impact as a product of expected value, cost, and risk rather than simply capabilities. For each of these, we

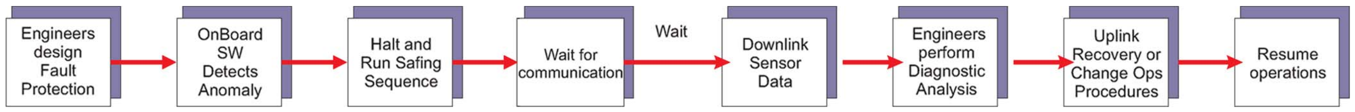
Engineers design Fault Protection
OnBoard SW Detects Anomaly
Halt and Run Safing Sequence
Wait for communication
Wait
Downlink Sensor Data
Engineers perform Diagnostic Analysis
Uplink Recovery or Change Ops Procedures
Resume operations

Fig. 1. Fault protection flow.

identify characteristics of MBD that hinder its adoption in the spacecraft domain using the Livingstone flight experience and hypothetical application of MBD to a more recent mission as examples. In the latter case, we propose an estimate for the ideal maximum value that MBD could provide for a mission, as well as a method for reasoning about the *expected* value of the technology.

Finally, in Section XI, we discuss a small set of systems that are model-based, or pertain to diagnosis and recovery, or have been fielded in operational use, with our ultimate interest being a system that is all three. In Section X, we revisit the question of potential impact by summarizing our own experience with the cost, risk, and value tradeoff for MBD, as well as our understanding of why in the spacecraft domain it has not enjoyed the same success as other model-based technologies.

II. Traditional FP Process

Before discussing MBD, we briefly discuss spacecraft FP for the purpose of comparison. The primary purpose of FP is to ensure that anomalies or operational problems encountered during operation of the spacecraft do not result in permanent reduction in the spacecraft's capabilities or loss of the mission itself. As Neilson [1] explains in an excellent overview of the FP system for the Mars Exploration Rovers (MER), FP is an engineering process that incorporates robustness to faults into spacecraft hardware, software, systems engineering, and operations. All of these systems are engineered to work together to reduce the likelihood that any reasonably plausible contingency will result in permanent loss of mission capabilities. This paper is largely concerned with onboard software for active fault detection, isolation, and recovery. We note, however, that the onboard system is just one aspect of the overall FP engineering process, and, in addition, the onboard system for protecting the spacecraft is typically a mix of hardware and software. Traditionally, the FP engineering process is driven by fault modes, effects, and criticality analysis (FMECA). This process typically determines the possible faults of a system or a subsystem, some notion of their likelihoods, and an analysis of the impact of each. If the likelihood of a fault is deemed sufficiently high and the impact is sufficiently negative, the analysis would also include how the fault would be detected and what the appropriate response would be.

The appropriate response to a fault may depend upon the phase of the mission. We use the term *critical phase* of a mission to mean periods of a mission where the spacecraft must take specific actions (a *critical sequence*), or loss or serious degradation of the mission will result. For example, during the entry, descent, and landing sequence, each MER entered the Martian atmosphere at over 10 000 mi/h. At specific times or altitudes, it jettisoned a protective shell, deployed airbags, and the like. Failure to execute a step in this 6-min sequence would end the mission. We say the system must *fail operational* in that any failure that occurs must be taken into account by the FP approach, for example, by switching between redundant subsystems, to allow execution to continue. Accordingly, during development of the spacecraft, an enormous amount of attention is given to the precise critical sequence the spacecraft will execute: which failures are likely, how they will be detected, and how they will be immediately mitigated. Since the response must be timely, it must be available onboard. This may involve something as simple as a table mapping observed sensor values to commands that should be issued in response, for example, to switch to a redundant backup. For very complex critical sequences on large spacecraft, a much more elaborate method of determining responses may be developed, such as for the Cassini orbital insertion at Saturn [2].

In a *noncritical phase*, it is still possible to damage or lose the mission due to a fault, but there is no added constraint of having to execute a critical sequence. Thus, typical FP systems tend to focus on mitigating or postponing the impact of the fault by changing the spacecraft's state or behavior. Fig. 1 illustrates a simplified FP process. In the first step, engineers use FMECA to identify situations dangerous to the spacecraft (e.g., low voltage on the power bus), how they would be detected, and how the spacecraft hardware, software, ground system, and operators should respond. The hardware and software portions of this policy are implemented on the spacecraft. In step 2, when a fault occurs, onboard software or, in some cases, hardware is triggered by onboard sensors and invokes the proscribed response. In step 3, if the policy maps the sensor readings to a serious problem, the spacecraft is often placed into a *safe mode*, where the only actions taken are those that maintain the spacecraft in a quiescent state until communication can be established in step 4. For example, on the MERs, draining the battery risks not being able to heat the rover during the cold Martian night and not being able to communicate with earth when expected. The system-level MER FP includes a hardware battery controller that disconnects the batteries should a serious hardware or software failure begin to drain them. In this case, the system's heaters, solar panels, and flight software work together to ensure that the rover does nothing but stay warm and wake once during the day to contact earth using very few hardware and software subsystems and minimal power. If the situation appears to be less grievous or more localized, then a less drastic response may be taken at first. On the MERs, subsystem behaviors incorporate subsystem-level FP [3]. If the rover's arm draws more than the allowed current during use, it is marked failed. The arm behavior ignores any subsequent requests to use the arm, and the rover driving behavior is disabled if the arm is not stowed away. Routine communication with earth and other tasks that do not involve the arm continue as normal. In step 5, during the next possibility for communication (hours to days, depending on the mission), sensor values from the spacecraft are downloaded, so ground operators can inspect the telemetry and debug the arm. In many cases, engineers may send commands to carefully gain more information about the spacecraft's condition or may attempt to reproduce the issue on an earth-bound copy of the spacecraft. In step 6, when the anomaly is understood, engineers may simply re-enable the spacecraft, upload a software patch to deal with the issue, or change operational procedures to allow the mission to work around it. A process similar to this has protected the MERs for over four years. A great summary of the anomalies and faults encountered by the rovers in the first 780 sols (Martian days) of operation is available [4].

The approach of safing the entire spacecraft or select subsystems when faults are suspected has the advantage that one need not know exactly which fault is causing the operational problem. If the rover arm exceeds an operational current limit, use of the arm ceases, which applies for a limitless number of reasons the arm might be malfunctioning from a motor short to a rock stuck in an actuator. Similarly, if the battery is being dangerously drained, the flight software may shut down its activities, but at some point, if the situation persists, the hardware will effectively turn off the rover and restart it in a fresh quiescent state when solar power is available.

The above FP approach is to carefully engineer the robust but minimal fail-operational capabilities needed for critical periods and, otherwise, to engineer hardware, software, and operations, so the spacecraft can be put in a safe state in response to plausible anomalies. This characterizes a wide range of FP systems that have been flown. The remainder of this paper is about experience gained attempting to improve upon this approach using MBD technology. The intent was to provide greater spacecraft autonomy and robustness, greater science return from missions, reduced operation costs, reduced analysis cost for a mission, and reduced flight software development cost. Section III describes Livingstone and L2, two model-based diagnosis systems that have been flown as experiments onboard the National Aeronautics and Space Administration (NASA) spacecraft, as well as demonstrated on a number of testbeds. Section III describes how Livingstone was expected to provide the benefits described above. Later, we describe why we believe these benefits were not realized, and provide some analysis of where we believe the problem lies.

III. Livingstone System

Livingstone [5], [6] is an MBD system in the style of a general diagnostic engine [7] and Sherlock [8]. By diagnosis, we mean the problem of observing a mechanical, software, or other system and determining what failures, if any, its internal components have suffered. By MBD, we mean diagnostic systems arising from computer science or artificial intelligence approaches, where a generic software inference engine is developed in the hope of addressing a large class of diagnosis problems [8], [9]. Later, system models that adapt the generic engine to a specific diagnosis domain are created. Typically, a model first describes the kinds of components that comprise the mechanical system and their individual behavior. For example, a valve may be open and allow fluid flow, or closed and not allow fluid flow. Livingstone also includes a model of how each component can be commanded between states. When in the open state, a valve can be commanded to the closed state. However, if a valve is in the stuck closed state, commanding it has no effect. Once the components are described, the model must specify how they are interconnected. For example, consider the Cassini main propulsion system in Fig. 2, which was used as a benchmark in the Livingstone papers. To avoid a persistent confusion, we note that Livingstone was run against a simulation of Cassini for demonstration purposes and not as a part of the Cassini mission. The system is operated by opening valves to allow pressure from the helium tank to pressurize the oxidizer tank and the fuel tank. If additional valves are opened, this pushes the fuel and the oxidizer to exactly one of the engines, where it is ignited to create thrust. Owing to Cassini's decade-long mission, the system has many redundant paths between the various tanks and a backup engine. Valves in parallel allow a second path in case a valve sticks shut. Valves in series allow a path to be turned off if a valve sticks open.

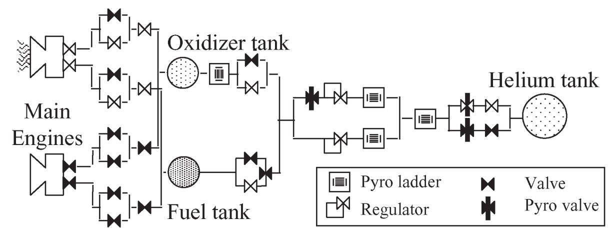

Fig. 2. Cassini main propulsion system.

When run against a simulation of this propulsion system, Livingstone's task was to use its model to monitor the commands given to the valves and infer if sensor readings (e.g., pressure at various points in the system) were consistent with the expected state of the valves. If not, Livingstone would infer the most likely set of failures (e.g., valves stuck open or closed) that explained the readings. One of the main innovations of Livingstone is its ability to use the same model and inference algorithm used for diagnosis to also infer what set of commands would cause the system to move to a state with a desired property. After a failure, Livingstone is able to determine the smallest number of valves to open or close to ensure that the oxidizer and the fuel could reach one of the engines. Thus, Livingstone could automatically manage the configuration of the propulsion system so that it always produced thrust when desired even in the face of multiple failures occurring over time.

Fig. 3 illustrates Livingstone's operation at an idealized schematic level. The generic Livingstone engine is given the high level model of the components of the spacecraft and their operation. During operation, Livingstone is fed with the stream of commands that are being given to the spacecraft and the readings from sensors embedded in the spacecraft hardware (e.g., switch status bits, temperature sensors, pressure transducers, etc.). Livingstone uses the model of the spacecraft's components and the command stream to predict the values of the sensors that should result from the commands assuming no components are failed. If there is a discrepancy between the predicted and observed sensors, then a failure is assumed. Livingstone uses the model of the components to simulate

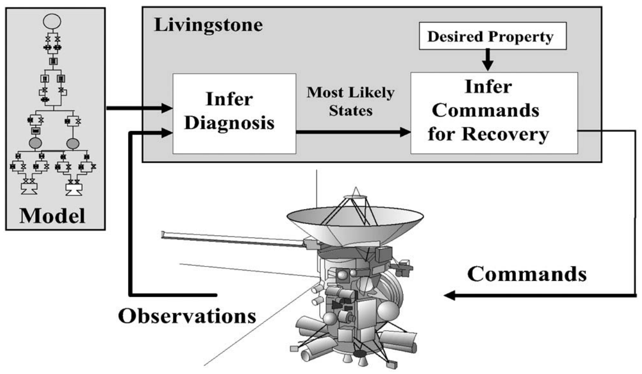

Fig. 3. MBD and recovery.

different combinations of failures. It uses a diagnosis algorithm to quickly focus on the most likely combination of failures that would predict the sensor values that are being observed. This combination of component failures is the diagnosis. To recover from a failure, Livingstone hypothesizes that a desired property is observed (e.g., the engine is receiving fuel). It uses the same predictive model and search algorithm to quickly infer commands (e.g., open sets of valves to route around a blockage) that would "predict" the desired property and, thus, mitigate the failure. Thus, it is capable of detecting, identifying, and also recovering from off-nominal conditions. With a model that includes commands, it can monitor execution of commands, detect anomalies, diagnose failures, and provide recovery actions for complex systems such as a spacecraft.

IV. Expected Benefits

The promise of a system like Livingstone flows from the concept that, in flight, it would use a model to infer the system-wide behavior given any combination of nominal and failed components and any sequence of commands. We envisioned a wide range of benefits if instead of manually reasoning through system-wide interactions under a fixed set of fault scenarios, we could simply describe the local behavior of components and allow an MBD system to infer the rest. Fig. 4 conveys where we expected the MBD process to deliver these benefits versus FP. In the first step, rather than predesigning responses for a fixed set of system-level failures, we describe the nominal and (perhaps multiple) failure behaviors of each type of component and how the components are interconnected. The user models that a valve can be open and allow fluid flow, or closed and stop fluid flow, or stuck open or stuck closed, and can arrange valve combinations far more complex than the Cassini model shown. Step 2 represents routine operation of the spacecraft with commands and sensor values monitored by Livingstone. In step 3, Livingstone would detect deviations from expected sensor values. In step 4, rather than waiting for intervention from the ground, Livingstone would immediately infer the state of the system even in the face of multiple valves sticking closed or open. In step 5, Livingstone would immediately perform the system-wide reasoning needed to start and stop fuel flow to the engines (or achieve any other state described by the model) from whatever state it had inferred for the system. Thus, the spacecraft would continue performing useful activities by finding alternative ways to perform them. We next describe some of the expected benefits in slightly more detail.

- *Reduced FP implementation costs*. Rather than develop a custom FP and safing system, the spacecraft would employ a reusable diagnosis engine that inferred the spacecraft's state and recovery responses on the fly from component models.
- *Significantly lower analysis costs for critical phases*. The fail-operational response needed during a critical phase would be inferred from the model online during the execution of the critical phase. Thus, we would reduce the manual analysis usually needed to determine how subsystems would interact during an anomaly and engineer a proper fail-operational response that could be statically encoded.
- *Greater robustness than traditional FP*. The onboard diagnoser would infer diagnoses of double and triple failures and suggest workarounds, where a traditional FP might include precomputed responses for single failures or certain critical anomalies involving multiple failures.
- *Greater mission return*. By inferring fail-operational responses for any state of the spacecraft, rather than manually engineering them for specific critical sequences, we could fail operational even in noncritical periods. During a fault, the MBD system would diagnose which component was failed. It would return to an operational state by reconfiguring redundant systems, resetting components, or canceling only activities involving failed components. Operations would resume with no need to wait for intervention from the ground, increasing productivity of the spacecraft.
- *Reduced mission staffing levels and operational costs*. Since the MBD system would resolve many anomalies without intervention from the ground, fewer expert engineers would be needed during spacecraft operations.

V. Flight Experience

A large number of talented people developed Livingstone and L2 applications for a wide variety of systems, which allowed us to gain the experience upon which this paper is based. Table I lists MBD applications developed for simulators, testbeds, and two flight experiments, along with the year and the number of different failures modeled. Where available, the number of person-months spent developing the Livingstone model and person-months developing the entire application (models, signal conditioning code, integration, etc.) is shown. Note that for the flight experiments, these numbers represent the additional cost of deploying Livingstone for a subset of the spacecraft systems. As discussed in subsequent sections, our observation is Livingstone performs a task somewhat orthogonal to FP, and these costs were not offset by any reduction in the development of an FP system.

Livingstone was chosen in 1996 to be a part of the Remote Agent spacecraft autonomy architecture [10]. In May 1999, it was flown on the DS-1 spacecraft as a technology experiment [11]. The first author gained direct experience with MBD technology by doing Livingstone development, modeling of the DS-1 spacecraft, and participating in mission operations during

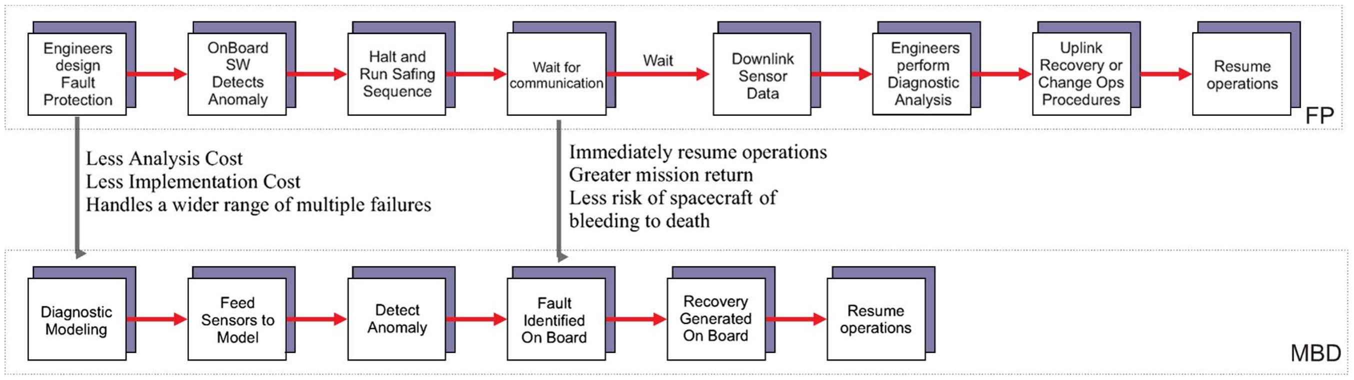
Engineers design Fault Protection
OnBoard SW Detects Anomaly
Halt and Run Safing Sequence
Wait for communication
Wait
Downlink Sensor Data
Engineers perform Diagnostic Analysis
Uplink Recovery or Change Ops Procedures
Resume operations
FP
Less Analysis Cost
Less Implementation Cost
Handles a wider range of multiple failures
Immediately resume operations
Greater mission return
Less risk of spacecraft of bleeding to death
Diagnostic Modeling
Feed Sensors to Model
Detect Anomaly
Fault Identified On Board
Recovery Generated On Board
Resume operations
MBD

Fig. 4. Expected benefits for MBD versus FP.

TABLE I
LIVINGSTONE AND L2 APPLICATIONS

Type	Domain	Subsystem		Year	System Effort Months	Model Effort Months	Types of Components
Simulator	Cassini	Propulsion	Liv.	1996			
Flight Exp	DS-1	Attitude control, switches	Liv.	1998	96	36	12
Testbed	ISRU	Chemical reactor	Liv.	1998	96	32	
Testbed	Interferometry	Optical bench	Liv.	1999			
Simulator	micro spacecraft		Liv.	2001			
Simulator	rover	Drive system	Liv.	2001			
Simulator	X-34	Propulsion	L2	2002			26
Simulator	X-37	Electronics	L2	2002	9	5	18
Flight Exp	Earth Orbiter 1	Instruments, cameras	L2	2003	12	2.8	

the experiment. Livingstone and the other components of the Remote Agent software were uploaded to DS-1's onboard computer via the Deep Space Network of radio antennas. The software was executed on the DS-1 flight computer for a 20-h test and a 6-h test. During these tests, the Remote Agent software ran on top of the existing DS-1 flight software, which included an FP system. The flight software fed live readings from the spacecraft's sensors to Livingstone to drive diagnosis. In addition, to ensure Livingstone would be exercised even if no failures occurred, Livingstone was fed with simulated sensor readings consistent with a set of four predetermined failure scenarios that had been modeled. These were failure of a switch position sensor, camera power switch stuck on, science instrument not responding, and thruster stuck closed. In the first case, Livingstone correctly ignored the sensor and, in the remaining cases, recommended recoveries of retrying the command, power-cycling the instrument, and switching thruster control modes, respectively.

Subsequently, we developed the follow-on L2 system, with several technical improvements [6]. With L2 and the associated modeling tools, our experience applying MBD and recovery systems to spacecraft greatly expanded. L2 was flown as a technology experiment on the EO-1 spacecraft, again operating on top of the existing FP system. L2 was activated on EO-1 for a total of 143 days in 2004 and 2005 and diagnosed 13 simulated failures [12]. Flight experiments for the X-34 and X-37 spacecraft were also developed, although those vehicles were never flown [13]. In addition to these flight experiments, Livingstone or L2 was applied to a Mars *in situ* propellant production testbed (ISRU) at the Kennedy Space Center [14], a testbed for the Space Interferometry Mission at the Jet Propulsion Laboratory, the Bio-Plex Mars habitat testbed at the Johnson Space Center, a testbed for the PSA micro spacecraft [15], and a rover testbed [16]. These applications tended to start with a set of known diagnostic scenarios and focus upon the relative ease or difficulty of causing the particular diagnosis formalism of Livingstone to diagnose them. While this is an interesting topic, it is beyond the scope of this paper. The Hyde system [17] is one of several systems that use a hybrid automata model and that describes the specific areas of expressivity it seeks to improve over Livingstone.

VI. ANALYSIS OF IMPACT

A great deal was learned from the efforts of the many talented teams that used Livingstone. In retrospect, however, it is fair to say that MBD did not transform or even impact how FP is done on NASA missions. To our knowledge, no NASA spacecraft has used Livingstone or any system like it as a part of its FP system, and as of this writing, there have been no additional flights beyond the research-funded DS-1 and EO-1 experiments. This begs the question of why we did not observe the wide range of benefits we expected from applying MBD and recovery to spacecraft operations, and why there was no "mission pull" and adoption of the technology.

It is tempting to explain the lack of penetration by imagining that missions are averse to incorporating new technologies. Certainly, revolutionary approaches are taken all the time. Consider the airbag landing system used on PATHFINDER and MER. Perhaps there is some hesitance to use unfamiliar or model-based software. Consider the case of planning and scheduling technology. Coincidentally, Livingstone flew with the HSTS

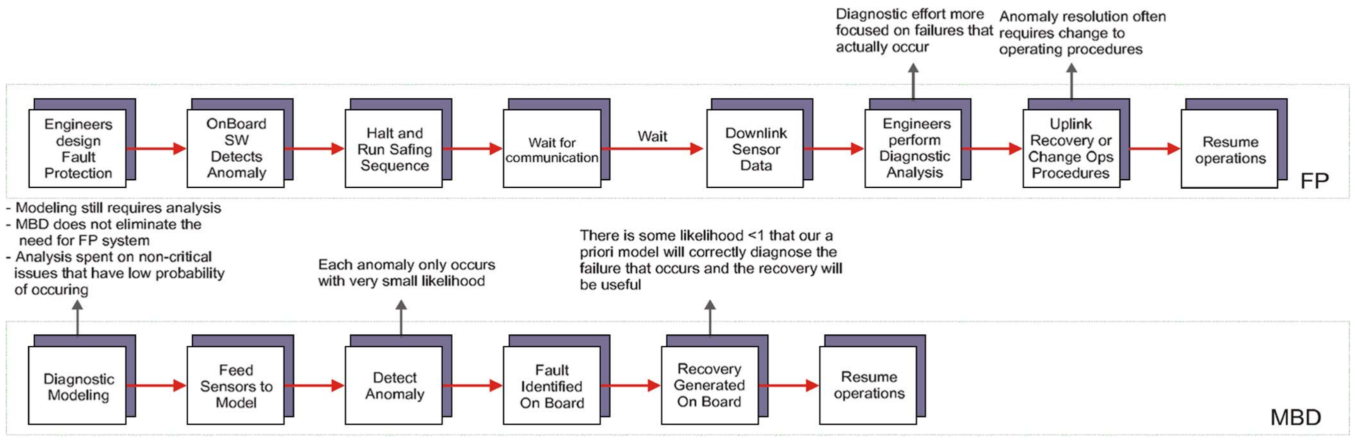
Diagnostic effort more focused on failures that actually occur
Anomaly resolution often requires change to operating procedures
Engineers design Fault Protection
OnBoard SW Detects Anomaly
Halt and Run Safing Sequence
Wait for communication
Wait
Downlink Sensor Data
Engineers perform Diagnostic Analysis
Uplink Recovery or Change Ops Procedures
Resume operations
FP
- Modeling still requires analysis
- MBD does not eliminate the need for FP system
- Analysis spent on non-critical issues that have low probability of occuring
Each anomaly only occurs with very small likelihood
There is some likelihood <1 that our a priori model will correctly diagnose the failure that occurs and the recovery will be useful
Diagnostic Modeling
Feed Sensors to Model
Detect Anomaly
Fault Identified On Board
Recovery Generated On Board
Resume operations
MBD

Fig. 5. Complications of providing value with MBD.

planner during the Remote Agent experiment, and L2 flew with the CASPER planner on EO-1. At a high level, planners are similar to Livingstone in that they comprise a generic inference engine and a model used to adapt it to a problem. A planner chooses actions to take to achieve a goal, rather than failures to explain a symptom, but the algorithms and the models are similar in the grand scheme of things. As described in Section XI, HSTS evolved into ground-based tools that have generated thousands of plans for the MERs and are scheduled for follow-on missions. CASPER became an operational tool on EO-1, continuing to run onboard for years, plan over 100 000 goals to date, and save millions of dollars in operating costs [18]. Thus, missions are willing to adopt new model-based software technologies.

As technology developers, there is a natural tendency to focus on the capabilities of a technology, such as MBD's ability to find diagnoses and make recoveries. The decision by a mission to adopt a technology is a combination of the expected value, cost, and risk given the characteristics of that particular mission. The three are intertwined. A mission may make a significant investment to buy down risk, as illustrated by the inclusion of a backup engine on Cassini. Conversely, a useful mission feature that provides value may be abandoned if the cost cannot be kept under control. In these terms, we believe it is relatively easy to understand the lack of penetration of MBD and recovery. We provide an overview before a detailed analysis.

Consider the noncritical phases of a mission and the proposal to add fail-operational capability. Fig. 5 illustrates some of the complications standing in the way of realizing the benefits as we imagined them. First, rather than eliminating analysis costs in step 1, our experience was that a diagnostic model requires the same type of FMECA as writing a traditional FP system, and in greater quantity. The additional analysis effort comes from the fact that engineers are forced to work out recovery strategies in detail beforehand, and once derived, all of these fault propagation, failure detection, and recovery analysis must be encoded in a single consistent model. We next note that spacecraft typically go into safe mode relatively rarely, intrinsically limiting our cost/benefit equation. In steps 3 and 4, we note that the value of a diagnosis and recovery capability captured by our models is only realized if a failure is correctly diagnosed and a recovery more productive than safing is achieved. Surveying real mission anomalies reveals that a few are failures that *a priori* we would have modeled and been able to recover.

Part of this is due to the enormous space of possible failures versus the limited budget and expressivity we have for developing a model. In addition, the strength of MBD lies in its use of the system design to infer diagnoses and recoveries. However, spacecraft typically are heavily analyzed and tested to ensure they perform robustly within the design parameters. When a failure occurs, it often arises from interactions between subsystems or the environment that were not explicitly captured in the design assumptions. In steps 5 and 6 of the FP process, we note that when failures do occur, operators typically need to carefully understand what has occurred and what assumptions about the spacecraft design have failed. They then upload commands to address the immediate situation and often make a change to operational rules for the spacecraft before proceeding. Thus, we will argue that the expected value provided by MBD during noncritical phases is not high, particularly when considered relative to the cost and risk involved.

Consider the critical phases of a mission and the proposal to use MBD and recovery to provide the required fail-operational capability. Typically, a critical phase is minutes to hours long, involves a known sequence of events, and is critical in the fate of a mission costing hundreds of millions of dollars. Added value and reduced cost from employing MBD have not been clearly demonstrated, but they would have to be incredibly high to offset the risk of not being able to verify in advance exactly how the spacecraft was going to respond to plausible scenarios during this short portion of the mission when an incorrect response will result in mission loss. Thus, we will argue that the risk of MBD during critical phases is high, and the value is unclear. The expected net value of using MBD over the life of a mission depends upon the expected value that the technology provides, minus the additional risks of using it and the cost of deploying it. In the next sections, we discuss the pressures on the expected value, cost, and risk that we believe hamper the use of onboard MBD and recovery systems. We support our analysis using the MER and DS-1 missions as examples.

VII. Low Expected Value

During Livingstone development, we focused upon the promise that Livingstone could provide value to a mission, for example, by potentially reducing the need for the spacecraft to enter a safe state. There was no attempt to quantify the magnitude of that value, leading us to overestimate it. In retrospect, in order for an MBD system to be exercised and provide value, a failure must occur, the MBD system must correctly diagnose the failure, and it must suggest a useful response. It is the likelihood of these events that will determine the usefulness of MBD. As with an insurance policy, the value actually provided by an MBD system during a mission is not known until after events have unfolded and the mission is complete. Unfortunately, we know of only two cases where an MBD system has been flown, which we discuss below. We then discuss methods for estimating the *expected* value if MBD were hypothetically deployed on a mission, and why we estimate it to be, in general, quite low.

During the two-day Livingstone flight experiment and the 143-day L2 flight experiment, none of the modeled failures of the spacecraft occurred. Two unmodeled failures in experiment-related software did occur and interfered with the flight experiment of Livingstone on DS-1. One failure we had not even considered modeling, and the other was explicitly declared out of scope during the modeling effort. Thus, the only two actual failures known to have occurred during a Livingstone flight experiment were not identified or recovered from [11]. The L2 flight experiment also generated 14 false positives, diagnosing modeled failures when none occurred. Thus, in these two very short experiments, this particular MBD system provided no operational value.

These two data points involving a particular MBD technology are too few to draw any conclusions about the actual value provided by MBD for spacecraft applications. However, like an actuary, we can compute the *expected* value that MBD would provide to other missions from the product of the likelihood of the events leading to MBD operating correctly and the value of it doing so. To estimate the expected value for a mission, we then need to estimate the following quantities:

- the likelihood of a failure actually occurring (≤ 1);
- the likelihood of correctly diagnosing the failure that actually occurs (≤ 1);
- the likelihood of providing proper response (≤ 1);
- the value of generating the correct diagnosis and response onboard (timeliness, reduced operation cost).

To provide a concrete example of how we would reason through this analysis for a mission, Table II summarizes Neilson's excellent overview of the anomalies in MER operations that occurred in the first two years of commanding two rovers on Mars. Note that Livingstone was not used on the MER mission; it is simply a well-known mission whose FP system and performance have been thoroughly described in the literature. Table II captures the anomalies that caused the two rovers to enter a fault response during approximately 1550 days worth of operations. The first column is a short name that we have given to the anomaly. The second denotes whether the problem was due to software, hardware, or interactions with the environment. We presume that most MBD applications do not apply to dynamic interactions with the environment, although this is incidental to our analysis. The third column denotes how many days passed between when the anomaly occurred and the subsystem involved in the anomaly was put back into routine operations. Note that, in many cases, other noneffected subsystems were put back into operation sooner. The final column describes what response was developed by the MER anomaly analysis team.

Table II captures anomalies during noncritical portions of the mission. In this case, we define the ideal value of having onboard diagnosis and recovery to be equivalent to the amount of operational time lost waiting in safe mode after an anomaly. This is the amount of time that would have been saved if every anomaly was resolved immediately. A second term in the expected value is the rate of failure within onboard spacecraft systems, which we expect to be quite low, in general. This is important because the product of our value term and this rate, conceptually, the expected amount of time the spacecraft spends in safe mode, is an upper bound on the value of onboard recovery, representing the ideal case where, for every anomaly, our MBD system produces a correct diagnosis and a productive recovery. This bound is simply the nature of attempting to provide value with onboard diagnosis and recovery.

Looking at Table II, the small number of hardware- and software-related anomalies suggests that the rate of anomaly occurrence is quite small. The amount of time spent on anomaly recovery was about 2% of the operational time, including a significant amount of time responding to environmental problems such as being stuck in soft sand. Thus, for MER, we know that the amount of potential operational time spent in safe mode turned out to be 2% of the operational time of the rover. This represents the ideal upper bound of the value of using MBD to immediately recover during routine operations if every anomaly could be correctly resolved autonomously onboard.

The reduction from ideal MBD performance to the expected value is determined by our estimates of the likelihood that a given MBD system will correctly diagnose these anomalies and the likelihood that it can provide a productive recovery. Looking at the failures in Table II, the first question to consider is out of the enormous space of component failure modes, what is the likelihood that *a priori* our models would have covered diagnoses and recoveries for these failures? The second question is what is the likelihood that our automated software would have produced recoveries that were both correct and more valuable than simply shutting down subsystems and waiting for ground analysis as the existing MER system does? Since MER did not use an onboard MBD system, discussing the likelihood of a correct diagnosis or recovery may be considered conjecture. We believe that one can get a sense of the (small) magnitude of these likelihoods from columns 3 and 4 of the table. These reflect the amount of time the engineers who built the rover, flight software, and operations system spent carefully probing the rovers with small experiments and then deriving a recovery and new operational policy that would accommodate the failure and the rover's interaction with the environment. We believe that the MER pattern of short infrequent bursts of extremely careful analysis indicates that the effective value of MBD would

TABLE II
MER ANOMALIES

Brief Description	Type	Days of Analysis	Recovery process
Wheel drive actuator	HW	4	Experiment to characterize capability of wheel Warm actuator before use Drive backward dragging wheel
Steering current	HW	4	Experiment to characterize source of current increase Use ’K’ turns to avoid steering failed wheel
Shoulder actuator current	HW	17	Experiment to characterize shoulder motor degradation Characterization of future arm failure on driving Change stowage policy to minimize thermal cycling & forestall failure
Heater stuck on	HW	*	Determine that survival heater was stuck on * Operations continued while problem was addressed Implement policy to remove batteries from power bus at night Rely on solar power to wake rover at dawn Trade off power savings vs. degradation of instrument due to cold
Late wakeup/ dust storm	Env	1	Solar panels woke up rover slightly late due to dust storm Rover missed time to start sequence, waited in standby mode Plan future sequences to start at least 1 hour after expected wakeup
Rock stuck in wheel	Env	7	Current spike explained by seeing rock in the wheel in imagery Several days of careful driving to dislodge rock
Stuck in sand	Env	40*	Imagery suggests rover is not moving, wheels 70% buried in sand * Continue all non-drive activities on Mars during analysis Set up testbed with similar consistency soil, practice escape strategies Carefully drive out using escape strategy Augment driving policy to avoid wheel embedding on future drives
Flash file system anomaly	SW	14	Overloaded file system table prevents creation of new files Rover continuously reboots Understand what on-board FP system is doing, what is causing reboots Send command to rover to start up without file system, gain control of rover Determine issue with file system. Clear and rebuild file system Carefully manage production of files Return to nominal ops. Later upload patch
Startup race condition	SW	2	Lose communication window every few hundred sols Added short keep out period after startup
Imaging race condition	SW	2	Imaging HW shut down while sequence still reading data from HW Shut down sequence fixed to halt imager sequence before HW shut down
Corrupt command	SW	6*	Solar conjunction test of corrupt commands overloads command handler *Normal commanding resumed after solar conjunction over
Variable eval exception	SW	4*	Same global defined in two sequences running in parallel, result in fault ∗ Includes idle weekend Do not run two scripts that define same global
Upload fault	SW	2	Initial uplink through orbiter experiment overloaded CPU Pad uplink file, limit size

be significantly less than adding 2% to the rovers’ operational time.

In general, we believe that the likelihood of failure is low for operational spacecraft, which means that even the potential benefit for MBD is limited. In addition, we consider the likelihood of correct diagnosis and recovery to be low, further lowering the expected value of MBD. MBD-friendly situations where combinatorics and propagation of information are the issue in generating diagnoses and recoveries, as was initially suggested by Cassini’s 27-valve propulsion system and its 2^{27} configurations, do not appear to be the driving problem in spacecraft FP and anomaly resolution. For the types of applications we are familiar with, failures that do cause loss of operational time are typically complex and unexpected and break the modeling abstractions used to develop the system.

Consider the failure of the Galileo antenna to deploy or the MER rebooting due to Flash problems. These types of problems typically require detailed analysis, creativity, and validation before returning the spacecraft to an operational state. Thus, for the missions that we have considered in detail (all unmanned deep-space missions), we would assign a low expected value to the use of onboard MBD and recovery. We next consider risk and cost.

VIII. NONTRIVIAL COST INCREASES

During Livingstone development, we projected that future missions would enjoy substantial savings by eliminating the need to write a traditional FP system or to perform the system-level analysis done to determine the correct response to critical mission-ending anomalies. Much of the work previously done through system-level analysis would be done through inference. Costs would largely consist of the effort needed to encode models for the MBD system, which would be reduced via graphical modeling tools developed with Livingstone.

We now believe that use of MBD adds significant analysis, development, and testing costs. In retrospect, use of an MBD system does not eliminate the need of an FP system and, thus, does not eliminate that source of analysis costs. Creating the models required for an MBD system also requires significant analysis specific to this usage. Unlike traditional approaches, the value proposition of MBD also promotes performing a significant amount of "speculative" analysis for a range of noncritical failures that do not strictly need to be completely analyzed *a priori*. Finally, our experience is that the envisioned MBD functionality adds a significant integration and testing effort to production of spacecraft flight software. We consider each of these factors, in turn, below.

We do not expect use of MBD to reduce the need for analysis related to FP because, in retrospect, diagnosis and FP are not equivalent. FP is a system engineering process that impacts the design of hardware, software, and operational procedures. It must ensure, for example, that if any hardware, software, environmental, or operational problem is draining the spacecraft's batteries, the combined hardware, software, operations system, and documented operator procedures have the maximum likelihood of stabilizing the situation before the vehicle is lost. This is a much broader problem than that of onboard component-level diagnosis and reconfiguration of the spacecraft. In addition, the ability to compute component-level diagnoses is often neither necessary nor sufficient to ensure spacecraft safety. The FP scheme typically identifies only faults (e.g., battery voltage is low) used to find a preplanned response that is meant to safe the spacecraft or continue a very specific critical sequence over a large space of problems induced by the hardware, software, or environment. For example, the FP design might place all nonessential devices on a separate power bus, which is simply turned off if there is any power-related anomaly. Thus, in the flight experiments, it was necessary to employ an FP system that did not need to do detailed diagnosis, while running Livingstone, which did not perform system FP. Since MBD as formulated does not address system FP, we did not see any demonstration that use of MBD technology eliminated or significantly reduced analysis costs for developing the basic FP capability missions require. In Section XI, we do discuss efforts to use MBD-related technologies in early design and validation of more traditional FP systems.

We imagined that future missions might use the ability of MBD to propagate behavior of individual components across a system model to generate recovery responses on the fly or before flight, which means the analysis costs of a mission using MBD would drop. Two issues are as follows: where do the models come from and what analysis do they eliminate? MBD requires a diagnostic model that is different than the simulation models used in routine spacecraft development. The DS-1 experience was that it is necessary to know the aspects of each component relevant to failure, the plausible failures to be modeled, how they manifest themselves locally, local actions that can be taken, and so on in order to scope and write a model that can be used for anomaly detection, identification, and recovery. Our experience was, therefore, that additional failure mode and effect analysis was needed to drive diagnostic modeling, rather than a model existing through some other process and then being used to replace traditional analysis.

The need for additional analysis is exacerbated by the proposition that an MBD system provides value by autonomously recovering from noncritical anomalies. Consider that the traditional FP strategy reserves the most detailed *a priori* analysis only for critical sequences and the process of safing the spacecraft. The majority of possible faults simply trigger the safing system without diagnosis to the component level. Since further analysis is performed *post hoc*, it is done only for those failures that actually occur and without the need to codify the diagnosis into a model. The main value ascribed to MBD was that it would do detailed diagnosis and recovery autonomously. This means that the analysis and modeling needed to diagnose and recover failures, and the nontrivial task of encoding those capabilities into software, must be done *a priori*, before we know which failures will occur. Thus, we perform the detailed analysis and modeling needed to automatically recover from many noncritical failures that most likely never occur and do not strictly require an immediate response in the unlikely event that they do.

Two overlooked additional costs are integrating the model with the spacecraft and testing. Signals generated by the spacecraft's internal sensors may need to be conditioned to remove transient disturbances or abstracted to match the diagnostic algorithm. Since we are attempting to use all available sensors to autonomously identify failures before abrupt system-level effects are seen, the Livingstone experience in this can represent at least as much work as modeling. Thus, we believe that integration costs are higher than for FP systems that typically focus on anomaly detection without isolation and use a smaller number of system-level measures. With respect to testing, one of the main characteristics of Livingstone is the ability to generate novel combinations of diagnoses and recoveries from the possible diagnoses and recoveries for individual components. However, it was still necessary to work through the possible failures, how the failure would propagate through the system, and how Livingstone would respond, and then validate the expected behavior through testing. Given the space of possible responses Livingstone could generate, the issue of how to validate it at a reasonable cost was an issue during experiments and remains an issue.

As a point of interest, Table I lists approximate development costs for Livingstone applications where it was possible to make an estimate. On the DS-1 experiment, three people worked part time for a total of approximately 36 person-months developing a model of five subsystems of the spacecraft. Approximately 96 person-months were spent, including the model and all of the integration and sensor signal conditioning necessary to run onboard the spacecraft. For the EO-1 experiment, a total of 2.8 person-months were spent modeling and approximately 12 person-months were invested in integration and signal conditioning. It is important to note that these figures are not the cost of the FP system, as Livingstone alone was unable to provide FP for the spacecraft. These are the costs to develop a diagnosis and recovery system for combinations of failures in specific subsystems.

In summary, we believe that the proposed use of MBD represents a cost increase rather than savings. The need to develop a separate set of component-level diagnostic models adds cost and does not significantly offset FP analysis costs. The desire to have, *a priori*, a system that can autonomously diagnose and recover a spacecraft even for noncritical anomalies appears to introduce additional detailed analysis, model encoding, signal conditioning, and testing that is not necessary if the spacecraft is simply put in a safe mode when possible. We believe that the approach is also at a cost disadvantage due to the more complex testing and verification requirements necessary to ensure that any of the additional diagnoses and novel recoveries that such a system might generate during autonomous operations would not endanger the spacecraft.

IX. Increased Risk

Initially, we imagined that MBD would reduce risk of mission loss by generating diagnoses and recoveries on the fly and increasing the range of situations over which specialized fault responses were available. Customers were instead concerned that the more complex response of an MBD system would itself cause an anomaly or compound a failure, and this would have an impact in terms of operational complexity during anomalies, lost science, or loss of mission capabilities. Reducing risk to a level acceptable to mission customers quickly emerged as an open issue that is key to future use.

We can divide sources of this perceived risk into three categories: increase in risk due to the fact that the MBD system is capable of a wider range of responses and generates them on the fly; increase in risk from continuing to operate after an anomaly when not strictly necessary; and the operational impact if the actions of an autonomous onboard recovery system must be understood and potentially mitigated when the spacecraft has ceased operating according to design and the ground team must intervene.

Missions considered that MBD would increase risk because of the increased complexity of the software's response and our inability to concisely characterize, enumerate, and validate the range of diagnoses and responses that the system might undertake. Rather than engineer a small number of safing responses that are as broadly applicable as possible, MBD seeks to generate recovery responses that are as specialized as possible on the fly. It is interesting to note that this does not necessarily imply that an MBD system responds to a broader range of anomalies, simply that it responds in a more specialized fashion to each. This means that there are far more variations in spacecraft response based on its state, and the full set of conditions and responses could not be enumerated and tested. In addition, the purpose of MBD is to propagate information across the modeled system to allow variations in response. Thus, it can be difficult to even concisely describe how small nonlocal variations in the space of inputs will impact the response, and difficult to argue that a specific set of test cases provides good coverage for validation. Combating this perceived increase in risk is a challenge since the ability of MBD to respond with a far wider range of behaviors than traditional FP is both its selling point and the source of concern that the system will do something unpredictable. Some work has been done to apply model-checking approaches [19] and to generate static representations that are simpler to validate [20], but this remains a significant issue for adoption.

An additional increase in risk comes from the proposal to continue to operate the spacecraft via a recovery generated onboard when it is possible to safe the spacecraft and await expert analysis. Consider the three wheel problems of Table II: 1) *wheel drive actuator*; 2) *rock stuck in the wheel*; and 3) *stuck in sand*. One can imagine misdiagnosing which failure was occurring and applying the recovery meant for another (e.g., attempting to drive in circles when stuck in sand rather than when a rock is in the wheel) could permanently trap or damage the rover. We do not have examples of Livingstone misdiagnosing a spacecraft failure, as none occurred during the flight experiments, but we do have examples of false positives (indicating a failure when none exists) during the EO-1 experiment. Thus, it is important to keep in mind that MBD may cause us to execute actions that are inappropriate for the true state of the spacecraft. This risk of exacerbating a failure through continued operation rather than safing is one of the items we are asking missions to trade against in order to gain the small expected value estimated in Section VIII.

Note that the impact is not just potential loss of mission. Having specialized generated responses to anomalies may make it harder and more costly to determine what exactly the spacecraft believes it is doing should something go wrong and mission controllers need to intervene. This is hard to characterize exactly, as we do not have good examples of a model-based FP system resolving an anomaly in operations, or of it needing assistance or intervention from the ground. We experienced a little of this in the Remote Agent experiment due to actual nondiagnosed anomalies during the experiment [21]. It may be better to consider how difficult it is to debug anomalies from millions of miles away even with (comparatively) simple safing and recovery actions, as in the Flash anomaly in the Spirit Mars rover [22]. For the Livingstone flight experiments, risk was acceptable because the MBD system operated on top of a complete separate FP system, which protected the spacecraft. On DS-1, for example, the flight software included an FP system that monitored the spacecraft for indications that any occurrence, including commands given by Livingstone, was putting the spacecraft in a risky situation. The FP system would then stop all commanding of the spacecraft and put the spacecraft into a safe mode to await contact from mission controllers. In addition, Livingstone's communication with the spacecraft was through a filter that ensured Livingstone could only send specific commands that had been analyzed for safety. If the underlying FP system were activated, that filter would be closed, and Livingstone would be terminated.

In an unforgiving environment such as space, Livingstone's ability to provide novel diagnoses and recoveries to failure combinations we had not explicitly considered was far less important than being able to verify exactly how it was going to respond in the most likely and most critical anomaly situations. Guarding an MBD system with a traditional FP system and restricting the commands it can give is one approach to bringing it to the level of predictability needed to convince mission stakeholders that the spacecraft will not go into an unsafe state. This strategy was appropriate for experiments whose purpose was to show that the technology could be flown. In routine operations, it would tend to undermine the cost and value arguments of using MBD. If MBD technology or, to an extent, any autonomous onboard technology is to make an impact on operations, we believe that this open question of predictability, validation, and risk must be addressed regardless of what the proposed value is.

X. Discussion

In this paper, we have discussed the expectations for MBD and recovery systems such as Livingstone and why we believe that not all of those expectations were met. We attempted to lay out the basic cost/benefit drivers in a domain of interest (unmanned spacecraft) and our understanding of why MBD and recovery have found relatively little traction. We can grossly characterize the common practices in FP to be identifying those contingencies where an active specific response to anomalies must be made (e.g., loss of a motor during an orbital insertion) and providing identification and response to those states. In other anomalous conditions, the spacecraft is safed, and engineers diagnose the problem *post hoc*. For the missions we have considered, this approach seems to provide lower risk and more than adequate value in terms of anomaly response when compared with MBD. In addition, we have not yet developed or seen an argument or demonstration that the total analysis, development, and testing cost for an MBD-based system would be lower than those based on common practice.

During noncritical mission phases, the net value of having onboard diagnosis and recovery is low since we are free to simply put the spacecraft into a safe mode and only then invest resources attempting to find a diagnosis or response. During critical phases (as well as noncritical), the real need to circumscribe and validate the responses of the FP system decreases the proposed value of MBD's ability to generate novel responses, while increasing its testing and analysis costs in an attempt to contain risk. We also believe that the key questions of how onboard component-level diagnosis fits in with and adds value to the broader task of FP engineering and how MBD technologies would reduce FP costs remain open. Thus, we believe it is difficult to justify the use of onboard generative diagnosis and recovery systems like those we have been involved with based on cost, risk, or value, at least for the type of missions with which we are familiar.

We can also summarize our analysis of MBD for spacecraft by noting that MBD and recovery are a "deep" strategy. For a given failure, we incur cost to automatically generate the most specific diagnosis and most complete recovery we can, relying heavily upon our diagnostic model, current sensor readings, and our inference algorithms. The design of an FP system, generally speaking, takes a "broad" strategy. We find a response involving the spacecraft hardware, software, and operations team that covers the maximum number of contingencies regardless of underlying cause, making the minimal number of assumptions about what systems on the spacecraft are operating as designed and restoring the minimal functionality needed for spacecraft survival. We can easily construct specific scenarios where, assuming the model holds, the response of an onboard MBD system is more powerful and even avoids loss of the spacecraft. However, in real operations where the likelihood of any particular failure is low, the space of possible failures is enormous, and the correctness of a model once failures occur is suspect, the broad approach is superior.

It is often proposed that very ambitious missions, such as a penetrator to melt through Europa's icy shell, will require advanced onboard autonomy in the style of MBD. This implies that the mission design is such that the credible risks of losing the spacecraft would be complex or numerous enough that they could not be reasonably characterized and a policy developed *a priori*, so that onboard inference of specialized responses would be necessary. For such a mission to be deemed feasible and worth a risk in the billions of dollars, and not simply deferred or redesigned, MBD would need to be seen as buying down an enormous amount of risk, rather than introducing it. This would require dealing with the issues of verification, characterizing coverage and behavior, and reasoning from a model that becomes incomplete during failures.

The ideal application for MBD technology would be a system where concise component-level models of nominal operation and failure modes are fairly easy to encode to reduce cost and are not likely to be grossly violated during failure (e.g., components not linked in the design do not influence each other under failure conditions), so the likelihood of reacting properly during failure is high. Although made of easy-to-model components, the system would need to have enough combinatorial complexity to allow MBD diagnosis algorithms to provide leverage. Two such areas might be modular robotics [23], where the functionality of a machine is achieved by reconfiguring tens or hundreds or thousands of identical very simple machines, or autonomic computing [24], i.e., the task of autonomously managing networks of communication and computation equipment, where there is a very strong abstraction between the functionality of a component of the network and the underlying hardware.

This is not to say there are not ways to make MBD more attractive to spacecraft missions. One theme is to use MBD to assist people with ground-based diagnosis and recovery. As with the Eureka system described in Section XI, one challenge is how an MBD system would assist domain experts in the kind of unanticipated anomalies that they find challenging. One potential answer is to move from a focus on automated diagnosis to mixed initiative (human-in-the-loop) use of diagnosis technology in both the development of broad FP systems and the process of diagnosis during an anomaly. By analogy, in mixed initiative planning [25] rather than an autonomous onboard system generating and executing a plan, planning software assists a human expert in developing a plan for later execution. The human expert remains the final authority on what is included in the plan and its validity. This approach has a raft of advantages for spacecraft operations [26]. Intuitively, it allows the planning software and model to perform 80% of the planning task it is capable of performing correctly. At the same time, cleaning up the planner's suggestions is far easier for the human expert than creating a plan without the planner's suggestions, analysis, and explanations. It frees us from what is potentially the fool's errand of attempting to develop models that are complete and correct under any circumstances. Finally, it involves the user in the process of developing a plan, so he or she is confident in what the plan does and why. By analogy, a mixed-initiative MBD system might assist a human expert in developing a broad FP system that can be concisely described, validated, or further modified by hand to reflect factors not captured by the MBD system or its model. Existing commercial systems for analyzing diagnosability and sensor placement are instances of this approach. Further moves in this direction include using MBD to compile FP rules that can be verified and modified [20] or to verify the fault response design of a traditional FP system [27]. We believe that this style of leveraging MBD technology represents an opportunity for the community to gain greater impact on spacecraft operations and other endeavors.

For potential customers of MBD or other similar technologies, we hope that this paper might inspire ways of thinking about the effective value to a project or mission. For advocates, we hope that this paper might gather common criticisms of model-based diagnosis technology into specific categories that can be addressed through research and development or rebutted through counterexamples or demonstration.

XI. Related Work

Here, we discuss techniques for analyzing the benefits of a technology similar to the one presented for MBD. We then present a small set of systems that are model based, or pertain to diagnosis and recovery, or have been fielded in operational use, our ultimate interest being finding a system that is all three. Finally, we discuss some systems that are using MBD-related technologies in a more mixed-initiative format for development of fault-protection systems.

A number of analytic approaches exist for examining the cost/benefit tradeoffs of technology investments. The cost/benefit analysis presented in this paper is an instance of a more general technology investment analysis described by Lincoln *et al.* [31]. This methodology has been applied to other spacecraft autonomy technologies, but not MBD in particular. Chase *et al.* [32] propose a utility-based method based on science return for evaluating inclusion of new technologies into Mars rover missions. Similar analyses have been applied to Integrated Vehicle Health Management (IVHM) technologies [28]–[30]. IVHM is broader and somewhat orthogonal to MBD and recovery as we have framed it, in that IVHM is typically concerned with all aspects of supporting operations of one or a fleet of systems. The focus of IVHM is typically on increasing operational availability and reducing maintenance and support costs, with somewhat less emphasis on online diagnosis of an operational system to allow it to continue a specific sortie or mission. Williams [28], for example, describes a discrete event simulation that can compute different quantitative measures of effectiveness such as missions completed or number of vehicles in maintenance in a given period. The tool is aimed at analyzing the sensitivity of various operational concepts to given assumptions about the performance of various IVHM capabilities. This paper suggests a very similar less mature but potentially more specialized method for evaluating MBD specifically. It also focuses on how assumptions about performance and the customer's operational model (the input to Williams' analysis methods) led to overestimating the impact of MBD and recovery, and what factors in hindsight might have allowed a more accurate cost/benefit analysis.

We next discuss operational diagnosis systems, a few model-based systems, and, finally, MBD systems. There are many commercially successful diagnosis applications. For example, every car sold in the U.S. since 1996 is required to comply with the Onboard Diagnostics II standard, which specifies a set of onboard tests and makes diagnostic information available to off-board diagnostic systems. Some vehicles are even able to perform some amount of active testing and have a "limp home" mode when sensors or engine control actuators are suspect [33]. However, what we are specifically interested in are systems that generate diagnoses online, based on some generic engine or set of principles plus a domain description of some sort, rather than systems where engineering analysis is encoded into static code or a table representing a recovery policy. It is interesting to note that when millions of the same unit will be manufactured (for example, a vehicle controller), one can amortize the cost of developing a high-performance control system through engineering analysis, somewhat undermining the cost reduction motivation of MBD.

The very successful Cassini mission, whose main propulsion system was later used as a benchmark problem in development of Livingstone, made use of a very capable rule-based fault diagnosis and recovery system in operations [2], [34]. That is, an FMECA process was used to derive the set of critical failures, the symptoms, or monitored sensor values that would result, and the appropriate responses. These mappings from monitored values to diagnosed states were encoded in rules. The appropriate commands to respond to each state were similarly encoded. For situations that could not be mapped directly from the sensor states to a diagnosis, the spacecraft would temporarily be set to a simple safe state, and then the spacecraft would execute a sequence of commands designed to reveal the problem or move the spacecraft from the safe state to an operational state.

There have been quite a few operational uses of what may be loosely called model-based autonomy technologies outside the arena of MBD. For example, the EO-1 spacecraft [18], 2003 Mars Exploration Rovers [35], 2007 Phoenix Mars Lander [26], and upcoming Mars Science Laboratory rover [26] all used or are preparing to use model-based planning and scheduling software in routine operations. Thousands of daily operational plans have been generated for the Mars Exploration Rovers, and Casper planner was the basis of a low-cost mission extension for the EO-1 spacecraft. We believe some of the differences in impact between the conceptually similar technologies of model-based planning and MBD can be put into the context of our analysis. First, the likelihood of the system being called into use is essentially 1.0 in the planning case, as the planner is typically in daily use to generate plans for routine operations. Contrast this with the diagnosis system, where the likelihood of providing utility is the product of the likelihood of each failure, the likelihood that the failure was *a priori* covered by the diagnostic model, and the likelihood that there is an effective recovery. Second, based on our experience with diagnosis and planning, the cost of modeling is lower for planning. The planner model concerns only nominal operation of the spacecraft, which is typically well understood, is often well documented, and, in some cases, can be translated into a model from existing mission artifacts [26]. Compare this with MBD, where the modeling task is to write a set of models that capture, again *a priori*, a set of relevant failures and how the failure signals are propagated across the system once it stops behaving according to its nominal model.

Researchers at Xerox PARC developed a model-based system for planning and scheduling print jobs within reconfigurable high Xerox end printers [36] that have been used in Xerox products since 1995. Engineers developing new variations of the machines write constraints between the sheets of paper and components in the printing process, and do not explicitly write a schedule or scheduling software for the machine. For each print job, a model-driven scheduler within the printer develops a schedule that is optimal for the job characteristics and the constraints imposed by the components available in the machine. This represents a commercially successful deployment of model-based reasoning, but does not appear to include any diagnosis, recovery, or handling of anomalies during operation of the machine.

We now come to MBD systems. Researchers at Xerox PARC developed an MBD system for copiers [37] intended to assist field technicians. After being presented to technicians, the system was not deployed, and a community knowledge-sharing system was deployed instead [38]. To paraphrase, the model-based diagnoser was not deployed because technicians knew how to identify and correct common faults, and small optimizations in that process were not of high value. The real issue was the unexpected issue that was not foreseen during the design of the machine or development of the diagnostic models. These might arise from operating the machine in extreme environments, unintentional interactions of components in a newly released design, unanticipated failure modes as the machines age, and so on. Thus, after a study of the technicians' work process, Eureka did not attempt to augment or replace the expert technicians' ability to perform diagnostic reasoning. Eureka instead allows technicians to exchange tips on new faults, diagnoses, and responses as they are created in the field, the success of which can be judged by its 20 000 users.

TEAMS [39] is a commercial design and diagnosis product that has been applied to a number of aerospace systems. TEAMS Designer allows the user to specify flow of the material or signals between subsystems of a physical system, specify sensor placement, and write tests on sensor values. TEAMS uses this information to perform offline testability analysis to determine which component failures can be detected and further isolated, and make recommendations to improve testability. This capability has been used for analysis of large aerospace systems, and researchers are investigating its use in determining the adequacy of FP responses very early in the design of spacecraft systems [27]. TEAMS-RT compiles the TEAMS model into a matrix that is used to determine which subsystems may be faulty from the outcome of sensor tests and which can be examined and validated before use. Researchers developed a TEAMS-RT prototype to demonstrate near real-time diagnosis of systems onboard a UH-60 Helicopter [40]. A prototype to assist engineers in making prelaunch diagnosis decisions for NASA's Ares 1-X launch vehicle is currently in development, as is a system for in-flight diagnosis of the NASA's Orion capsule [41]. These applications have the strong advantage that the diagnosis system is compiled into a matrix that can be validated, and the diagnostic output is interpreted and vetted by an engineer before action is taken. Similar work has been done to convert an MBD model to a set of rules that can be validated [20]. Challenges always arise when attempting to move from a prototype to a production system with demands on risk, cost, and benefits. We believe that the best opportunity lies in these kinds of applications where MBD technology assists in developing a broad FP system that can be validated before execution, and, where appropriate, the output is vetted by an expert before being acted upon.

Acknowledgment

The inspiration, original design, and implementation of Livingstone are due to B. Williams and P. Nayak. P. Nayak and J. Kurien developed L2, which is maintained by L. Brownstone at the NASA Ames Research Center. The authors would like to thank G. Dorais, S. Poll, P. Robinson, A. Sweet, S. Hayden, L. Fesque, A. Patterson-Hine, T. Starbird, and particularly the three anonymous reviewers for valuable feedback that improved this paper. Livingstone and L2 applications were created by many talented people, including A. Bajwa, S. Christa, D. Clancy, P. Engrand, S. Feagan, C. Goodrich, M. Haddad, D. Hammond, S. Hayden, T. Hodge, B. Hudson, C. Ihlefeld, W. Larson, D. Lueck, C. Mattson, B. McLaughlin, B. Millar, P. Mullenix, C. Parrish, C. Pecheur, S. Pepke, C. Plaunt, S. Poll, G. Smith, A. Sweet, M. Schwabacher, W. Taylor, R. Washington, and S. Waterman.

References

[1] T. C. Neilson, "MER on-board surface fault protection," in *Proc. IEEE SMC*, Big Island, HI, Oct. 2005, pp. 14–19.
[2] G. Brown, D. E. Bernard, and R. D. Rasmussen, "Attitude and articulation control for the Cassini spacecraft: A fault tolerance overview," in *Proc. AIAA/IEEE Digital Avionics Syst. Conf.*, 1995, pp. 184–192.
[3] G. E. Reeves, "An overview of the Mars Exploration Rovers' flight software," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2005, vol. 1, pp. 1–7.
[4] J. Matijevic and E. Dewell, "Anomaly recovery and the Mars Exploration Rovers," *Prog. Astronaut. Aeronaut.*, vol. 220, pp. 547–568, 2007.
[5] B. C. Williams and P. P. Nayak, "A model-based approach to reactive self-configuring systems," in *Proc. AAAI*, 1996, pp. 971–978.
[6] J. Kurien and P. P. Nayak, "Back to the future with consistency based trajectory tracking," in *Proc. 17th Nat. Conf. AI, 12th Conf. Innovative Appl. AI*, Austin, TX, 2000, pp. 370–377.
[7] J. de Kleer and B. C. Williams, "Diagnosing multiple faults," *Artif. Intell.*, vol. 32, no. 1, pp. 97–130, Apr. 1987, reprinted in [42].
[8] J. de Kleer and B. C. Williams, "Diagnosis with behavioral modes," in *Proc. IJCAI*, Detroit, MI, 1989, pp. 1324–1330.
[9] R. Reiter, "A theory of diagnosis from first principles," *Artif. Intell.*, vol. 32, no. 1, pp. 57–95, Apr. 1987, reprinted in [42].
[10] N. Muscettola, P. P. Nayak, B. Pell, and B. C. Williams, "Remote agent: To boldly go where no AI system has gone before," *Artif. Intell.*, vol. 103, no. 1/2, pp. 5–47, Aug. 1998.
[11] D. E. Bernard, G. A. Dorais, C. Fry, E. B. Gamble, Jr., B. Kanefsky, J. Kurien, W. Millar, N. Muscettola, P. P. Nayak, B. Pell, K. Rajan, N. Rouquette, B. Smith, and B. C. Williams, "Design of the remote agent experiment for spacecraft autonomy," in *Proc. IEEE Aerosp. Conf.*, Aspen, CO, 1998, pp. 259–281.
[12] S. Hayden and A. Sweet, "Livingstone model-based diagnosis of earth observing one," in *Proc. AIAA 1st Intell. Syst. Tech. Conf.*, Chicago, IL, Sep. 2004.
[13] W. Maul, A. Chicatelli, C. Fulton, E. Balaban, A. Sweet, and S. Hayden, "Addressing the real-world challenges in the development of propulsion IVHM technology experiment (PITEX)," in *Proc. AIAA 1st Intell. Syst. Tech. Conf.*, Chicago, IL, Sep. 2004.
[14] W. Larson and C. Goodrich, "Intelligent systems software for human Mars missions," in *Proc. Int. Aeronaut. Congr.*, Rio de Janeiro, Brazil, 2000.

[15] G. A. Dorais, S. D. Desiano, Y. Gawdiak, and K. Nicewarner, "An autonomous control system for an intra-vehicular spacecraft mobile monitor prototype," in *Proc. 7th Int. Symp. Artif. Intell., Robot., Autom. Space*, Nara, Japan, 2003.

[16] J. L. Bresina, K. Golden, D. E. Smith, and R. Washington, "Increased flexibility and robustness for Mars rovers," in *Proc. 5th Int. Symp. Artif. Intell., Robot., Autom. Space*, 1999, pp. 175–184.

[17] S. Narasimhan and L. Brownston, "Hyde—A general framework for stochastic and hybrid model-based diagnosis," in *Proc. 18th Int. Workshop Principles Diagnosis*, Nashville, TN, May 2007, pp. 162–169.

[18] S. Chien, R. Sherwood, D. Tran, R. Castano, B. Cichy, A. Davies, G. Rabideau, N. Tang, M. Burl, D. Mandl, S. Frye, J. Hengemihle, J. D'Agostino, R. Bote, B. Trout, S. Shulman, S. Ungar, J. Van-Gaasbeck, D. Boyer, M. Griffin, H. Burke, R. Greeley, T. Doggett, K. Williams, V. Baker, and J. Dohm, "Autonomous science on the EO-1 mission," in *Proc. 7th i-SAIRAS*, Nara, Japan, 2003.

[19] A. Cimatti, C. Pecheur, and R. Cavada, "Formal verification of diagnosability via symbolic model checking," in *Proc. IJCAI*, Acapulco, Mexico, 2003, pp. 363–369.

[20] B. C. Williams, M. D. Ingham, S. H. Chung, and P. H. Elliott, "Model-based programming of intelligent embedded systems and robotic space explorers," *Proc. IEEE*, vol. 91, no. 1, pp. 212–237, Jan. 2003.

[21] P. Nayak, D. Bernard, G. Dorais, E. B. Gamble, Jr., B. Kanefsky, J. Kurien, W. Millar, N. Muscettola, K. Rajan, N. Roquette, B. Smith, W. Taylor, and Y. Tung, "Validating the DS-1 remote agent experiment," in *Proc. 5th i-SAIRAS*, 1999, p. 349.

[22] G. E. Reeves and T. C. Neilson, "The Mars rover spirit FLASH anomaly," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, 2005, pp. 4186–4199.

[23] M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, "Modular self-reconfigurable robot systems: Challenges and opportunities for the future," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 43–52, Mar. 2007.

[24] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Comput.*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[25] M. Burstein and D. McDermott, "Issues in the development of human-computer mixed-initiative planning," *Cognitive Technology: In Search of a Humane Interface*, B. Gorayska and J. L. Meyer, Eds., Amsterdam, The Netherlands: Elsevier Science B.V., 1996, pp. 285–303.

[26] A. Aghevli, A. Bachmann, J. Bresina, K. Greene, B. Kanefsky, J. Kurien, M. McCurdy, P. Morris, G. Pyrzak, C. Ratterman, A. Vera, and S. Wragg, "Planning applications for three mars missions with ensemble," in *Proc. 5th Int. Workshop Plan. Scheduling Space*, Baltimore, MD, Oct. 2006.

[27] T. Kurtoglu, S. B. Johnson, E. Barszcz, J. R. Johnson, and P. I. Robinson, "Integrating system health management into the early design of aerospace systems using Functional Fault Analysis," in *Proc. Int. Conf. Prognostics Health Manage.*, Denver, CO, 2008, pp. 1–11.

[28] Z. Williams, "Benefits of IVHM: An analytical approach," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, 2006.

[29] G. J. Kacprzynski, M. J. Roemer, and A. J. Hess, "Health management system design: Development, simulation and cost/benefit optimization," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, 2002, vol. 6, pp. 3065–3072.

[30] C. Hoyle, A. Mehr, I. Y. Tumer, and W. Chen, "On quantifying cost-benefit of ISHM in aerospace systems," in *Proc. ASME Int. Des. Eng. Tech. Conf.*, Las Vegas, NV, 2007, pp. 1–7.

[31] W. P. Lincoln, A. Elfes, T. Huntsberger, G. Rodriguez, and C. R. Weisbin, "Relative benefits of potential autonomy technology investments," in *Proc. Conf. Space Mission Challenges Inf. Technol.*, Pasadena, CA, 2003.

[32] J. P. Chase, A. Elfes, W. P. Lincoln, and C. R. Weisbin, "Identifying technology investments for future space missions," in *Proc. AIAA*, Long Beach, CA, 2003.

[33] C. O. Probst, *How to Understand, Service, and Modify Ford Fuel Injection and Electronic Engine Control*. Cambridge, MA: Robert Bentley, 1993.

[34] J. Hackney, D. E. Bernard, and R. D. Rasmussen, "The Cassini spacecraft: Object oriented flight control software," in *Proc. Guid. Control Conf.*, Keystone, CO, 1993.

[35] J. Bresina, A. Jonsson, P. Morris, and K. Rajan, "Activity planning for the Mars Exploration Rovers," in *Proc. 14th Int. Conf. Autom. Plan. Scheduling*, Monterey, CA, 2005.

[36] M. P. J. Fromherz, D. G. Bobrow, and J. de Kleer, "Model-based computing for design and control of reconfigurable systems," *AI Mag.*, vol. 24, no. 4, pp. 120–130, 2003.

[37] D. G. Bell, D. G. Bobrow, B. Falkenhainer, M. Fromherz, V. Saraswat, and M. Shirley, "RAPPER: The copier modeling project," in *Proc. 8th Int. Workshop Qualitative Reason. About Phys. Syst.*, Nara, Japan, 1994.

[38] D. G. Bobrow and J. Whalen, "Community knowledge sharing in practice: The Eureka story," *Reflections, J. Soc. Organizational Learn.*, vol. 4, no. 2, pp. 47–59, 2002.

[39] C. Deb, K. R. Pattipati, V. Raghavan, M. Shakeri, and R. Shrestha, "Multi-signal flow graphs: A novel approach for system testability analysis and fault diagnosis," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 10, no. 5, pp. 14–25, May 1995.

[40] A. Patterson-Hine, W. Hindson, D. Sanderfer, S. Deb, and C. Domagala, "A model-based health monitoring and diagnostic system for the UH-60 helicopter," in *Proc. Americal Helicopter Soc. 57th Annu. Forum*, Washington, DC, 2001.

[41] M. Schwabacher and R. Waterman, "Pre-launch diagnostics for launch vehicles," in *Proc. Aerosp. Conf.*, Big Sky, MT, 2008, pp. 1–8.

[42] W. Hamscher, L. Console, and J. de Kleer, *Readings in Model-Based Diagnosis*. San Mateo, CA: Morgan Kaufmann, 1992.

James Kurien received the B.S. and M.S. degrees in computer science from Rensselaer Polytechnic Institute, Troy, NY, and the M.S. and Ph.D. degrees from Brown University, Providence, RI.

From 1996 to 2001, he was with the Model-based Autonomy Group, National Aeronautics and Space Administration (NASA) Ames Research Center, Moffett Field, CA. He contributed to Livingstone and served on the development and flight operations teams for the Remote Agent experiment that tested Livingstone, a planner, and an executive in an experiment on the DS-1 spacecraft. He led the design and development of L2 which was subsequently flown as an experiment on the EO-1 spacecraft and demonstrated on testbeds for reusable launch vehicles. From 2001 to 2003, he was a member of the research staff with Xerox PARC and a Coinvestigator on a Defense Advanced Research Projects Agency project related to distributed sensor networks. In 2003, he returned to NASA to focus on mission development. He led the development team for the tactical planning system for the 2007 Phoenix Mars Lander and served as a Cognizant Engineer for the extension of the tool for the 2011 Mars Science Laboratory mission. Aspects of the system are also in daily operational use for the Mars Exploration Rovers and are being deployed for the International Space Station. Balancing technology development and mission operations, he also led the Planning and Scheduling Research Group with the NASA Ames Research Center, was a Ground Segment Manager for the NASA's Kepler mission, and served on review boards for a number of other missions. In 2010 he joined the Jet Propulsion Laboratory as a Group Supervisor in the Planning and Execution Division and Lead of the Activity Planning and Sequencing Subsystem for the 2011 Mars Science Laboratory rover mission.

María Dolores R-Moreno received the M.S. degree in physics from the Universidad Complutense de Madrid, Madrid, Spain, and the Ph.D. degree in computer science from the Universidad de Alcalá, Alcalá de Henares, Spain.

She previously worked as an Associate Professor and as a consultant. She spent one year with the National Aeronautics and Space Administration (NASA) Ames Research Center as a Postdoctoral Researcher and nine weeks as a Research Visitor with the European Space Agency's (ESA's) European Space Research and Technology Centre. She is currently an Associate Professor with the Departamento de Automática, Universidad de Alcalá. She is actively collaborating in ESA projects and participating with research groups at NASA. She has publications in journals such as *AI Magazine* and *Expert Systems With Application*. Her research focuses on automated AI planning and scheduling, monitoring and execution applied to real applications (i.e., aerospace, e-learning, or the Web), and genetic programming.

Dr. R-Moreno has served in the program committee of several international AI conferences and a Reviewer of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. She is a member of the editorial board of the *International Journal of Computer Science and Applications* and the International Association of Engineers, and of the Experts Group of the Editorial Idea Group, Inc. She has publications in the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING.