

Advanced Diagnostic System on Earth Observing One

Sandra C. Hayden^{*} Adam J. Sweet[†]
Ames Research Center, QSS Group, Inc, Moffett Field, CA, 94035, USA

Scott E. Christa[‡]
Ames Research Center, Aerospace Computing, Inc., Moffett Field, CA, 94035, USA

Daniel Tran[§]
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109-8099, USA

Seth Shulman[¶]
Goddard Space Flight Center, Honeywell, Greenbelt, MD, 20771, USA

In this infusion experiment, the Livingstone 2 (L2) model-based diagnosis engine, developed by the Computational Sciences division at NASA Ames Research Center, has been uploaded to the Earth Observing One (EO-1) satellite. L2 is integrated with the Autonomous Sciencecraft Experiment (ASE) which provides an on-board planning capability and a software bridge to the spacecraft's 1773 data bus. Using a model of the spacecraft subsystems, L2 predicts nominal state transitions initiated by control commands, monitors the spacecraft sensors, and, in the case of failure, isolates the fault based on the discrepant observations. Fault detection and isolation is done by determining a set of component modes, including most likely failures, which satisfy the current observations. All mode transitions and diagnoses are telemetered to the ground for analysis. The initial L2 model is scoped to EO-1's imaging instruments and solid state recorder. Diagnostic scenarios for EO-1's nominal imaging timeline are demonstrated by injecting simulated faults on-board the spacecraft. The solid state recorder stores the science images and also hosts the experiment software. The main objective of the experiment is to mature the L2 technology to Technology Readiness Level (TRL) 7. Experiment results are presented, as well as a discussion of the challenging technical issues encountered. Future extensions may explore coordination with the planner, and model-based ground operations.

I. Introduction

GENESIS project manager, Don Sweetnam, on the Genesis crash in September 2004 after parachutes failed to open: "Keep in mind that when we buttoned the system up at Kennedy Space Center and launched it in 2001, its fate was sealed," he said. "There was really nothing we could do at this stage to change the outcome." NASA has recently experienced a string of such failures: Columbia, Mars Polar Lander, Mars Climate Orbiter and Mars Observer. The common theme is that our current space systems have limited capability to recognize when the mission is in danger and recover to save the day. What is called for is a paradigm shift to a new strategy recognizing that mission-critical systems need on-board decision support, especially when operating without human oversight.

The technological approach of autonomy has arisen to resolve this challenge. The high-level requirements for autonomous systems may be specified as including:

- ▲ the capability to detect anomalous conditions and isolate to the root cause fault (*aka* diagnosis);

^{*} PI/PM, Computational Sciences Division, NASA Ames Research Center, M/S 269-3, AIAA Professional Member.

[†] Research Engineer, Computational Sciences Division, NASA Ames Research Center, M/S 269-1, AIAA Professional Member.

[‡] Systems Engineer, NASA Ames Research Center, M/S 247-2.

[§] ASE Software Lead, Artificial Intelligence Group, NASA Jet Propulsion Laboratory, M/S 126-347.

[¶] EO-1 Flight Operations Team Technical Lead, Flight Control Systems, NASA GSFC, M/S 428-2.

▲ the capability to plan and execute actions to enable mission continuation despite failures (*aka* recovery).

Over the past several years, NASA Ames Research Center (ARC) has been developing autonomous systems with these requirements in mind, and is establishing a track record of mission experience. In 1999, the autonomous Remote Agent Experiment (RAX) flew on the Deep Space One (DS-1) spacecraft [3] with an earlier version of Livingstone. Livingstone is a model-based diagnostic engine developed at Ames by the Model-Based Diagnosis and Recovery (MBDR) group [2]. Since then, the MBDR group has created the next version of Livingstone, called Livingstone 2 (L2). L2 was further developed by Integrated Vehicle Health Management (IVHM) applications: the Propulsion IVHM Technology Experiment for X-Vehicles (PITEX) project [7], performing monitoring and diagnosis of a high-fidelity simulation of the X-34 Main Propulsion System on flight-like hardware; and on the X-37 IVHM experiment where L2 memory management and coding was modified to meet Boeing’s stringent flight standards.

This paper describes the results of a year-long project in which L2 was uploaded to the Earth Observing One (EO-1) satellite to conduct diagnostic tests. EO-1 was developed by NASA Goddard Space Flight Center (GSFC) under the New Millennium Program (NMP), and launched in November 2000. EO-1 is an active earth science observation platform, operated by Goddard. In this technology infusion experiment, L2 and the spacecraft diagnostic models were integrated with the Autonomous Spacecraft Experiment (ASE) [4]. ASE, another NMP project, was developed at NASA Jet Propulsion Laboratory (JPL), and first ran on-board EO-1 on September 20, 2003. The autonomy software consists of JPL’s Continuous Activity Scheduling, Planning, Execution and Replanning (CASPER) planner; the science event detection software and the Spacecraft Command Language (SCL), developed by Interface and Control Systems (ICS). SCL provides an executive, a database and visibility to commands issued to the spacecraft, and observations of the spacecraft response telemetry on the 1773 data bus. L2 provides a diagnosis component to ASE, not included before.

Two major challenges were encountered during the course of this work: first, the development of a streamlined real-time interface capability to facilitate experiment integration with a real-time system; and second, integration and testing on the flight hardware testbeds. Incremental testing was utilized to build up the system in verifiable pieces, and automated regression testing ensured that new functionality was thoroughly verified before being baselined. The experiment was developed and deployed within a year under tight resource constraints which did not allow for model development to the full extent possible. Several interesting subsystems are not in scope: the attitude control subsystem (ACS) exhibits continuous behavior; the power subsystem interacts with many other client subsystems and serves as a good illustration of the value added by system health management in resolving root cause faults. A few component faults have occurred on the spacecraft; modeling these components and triggering these faults could allow L2 to diagnose actual faults rather than simulations. Software diagnosis is a relatively untouched research area which could also be explored here; complex autonomous systems are a fertile testbed for run-time software validation technologies. The nature of hardware and software models is essentially the same; real-time behavior is captured in the L2 model, abstracted from the underlying implementation.

In early September 2004, the combined L2 and ASE software was uploaded to EO-1. After successful checkout procedures, full scenario validation commenced. As of this writing, eight out of the seventeen defined scenarios have been validated, with the remaining scenarios to be completed in several weeks. Performance of the experiment has been flawless with no issues or technical problems.

II. Diagnostic Requirements for the Experiment

The primary goal of this experiment is to increase the Technology Readiness Level (TRL) of L2 by demonstration on a flight vehicle. However, several significant functional advances in the technology over previous work with Livingstone are to be demonstrated over the course of the experiment; these are shown in Table 1.

Table 1: L2 on EO-1’s intended functionality compared with previous Livingstone experiments

Functionality	Remote Agent (L1)	PITEX (L2)	L2 on EO-1
Spacecraft Hardware in the Loop	Yes	No	Yes
Multiple Hypotheses	No	Yes	Yes
Multiple Hypotheses with Backtracking	No	Yes	Yes
Diagnosis During Transients	No	Yes	Yes
Separation of Code and Model	Yes	No	Yes
Number of diagnostic scenarios	2	24	17
Long-term space operations	No	No	Yes

Remote Agent was a flight experiment on DS1, using the original Livingstone (L1). The PITEX experiment applied L2 to the proposed X-34 vehicle, running the diagnosis system on flight-like hardware with simulated data. As seen from the table, the L2 on EO-1 experiment will incorporate all previously developed functionality of L2, and demonstrate additional functionality as well. A subset of these requirements forms the success criteria for the experiment, as explained in the next section. Coverage of a number of diagnostic scenarios and the goal of long-term space operations (for days or weeks) are desired features rather than minimum requirements.

A. Minimum Success Criteria

In order for the experiment to be deemed successful, a minimum set of features are validated by on-board demonstration. These required features are the Minimum Success Criteria (MSC), identified below. Each MSC must be demonstrated with at least one scenario. In the results section, a scenario which demonstrates each of these requirements is presented.

- 1) Spacecraft Hardware in the Loop
 - ▲ The L2 experiment shall be deployed on-board EO-1, and shall demonstrate monitoring of nominal operations and diagnosis of anomalies in the spacecraft subsystems.
- 2) Multiple Hypotheses
 - ▲ Multiple alternative fault candidates shall be presented in the failure diagnosis, with an indication of relative likelihood.
- 3) Multiple Hypotheses with Backtracking
 - ▲ In light of new evidence, the list of diagnostic fault candidates shall be revised. This may entail a revision of the most likely fault candidate.
- 4) Diagnosis During Transients
 - ▲ Diagnosis of a failure shall not be delayed by concurrent commanding of the spacecraft. The real-time interface has the capability to diagnose subsystems prior to system-wide quiescence, in the face of concurrent, overlapping commanding. In other words, the state of each component is tracked regardless of whether commands are being simultaneously issued to several components. This allows diagnosis earlier than would be possible if we had to wait until all commands had been processed. This is explained further in section D on 'Diagnosis in Real-World Systems'.
- 5) Separation of Code and Model
 - ▲ All L2 code shall be independent of the diagnostic model. The idea of the model-based approach is to maintain a separation between the diagnostic engine and the model. In the PITEX project, this separation was not enforced in the Real-Time Interface (RTI), which contained domain-specific information in order to perform diagnosis while some subsystems are transitioning. As a result, any model change would require corresponding updates to the source code, and upload of the entire ASE/L2 code to the spacecraft, which takes over a week.

B. Diagnostic Scope of L2 Models

The scope of the EO-1 L2 model is a subset of the spacecraft components most relevant to the science data collection sequence: the two imaging instruments, called the Hyperion Science Instrument (HSI) and the Advanced Land Imager (ALI), and the solid state data recorder, called the Wideband Advanced Recorder Processor (WARP). To facilitate the integration of L2 with ASE, the model was scoped to utilize the commands and telemetry already made available to ASE by SCL.

Scenario scope is based on the nominal imaging sequence or Data Collection Event (DCE), the commands and telemetry observations sent to the Hyperion, ALI, and WARP. This sequence is:

- Components set to image collection mode
- Dark calibration image taken
- ALI and Hyperion aperture covers opened
- Earth image taken
- ALI and Hyperion aperture covers closed
- Dark calibration image taken again
- Components set to standby mode

Diagnostic scenarios were defined in order to get full coverage of fault modes in the model. Each of the fault scenarios is based on the nominal scenario, with only the minimal required telemetry modified to inject the fault. There are seventeen scenarios in total:

- one for the nominal data collection event,
- one for a dual nominal data collect (two successive earth images),

- and one to test each of the fault modes in the L2 model (15 faults in all).

The L2 models of EO-1 were created at Ames in an iterative four step process: knowledge acquisition, scope definition, model creation, and model testing. Knowledge about the components, and how they behave under nominal and fault conditions, was acquired from EO-1 engineers with several years of experience operating the spacecraft. Most of the EO-1 telemetry observations used by the model are already discrete, hence creating a discrete L2 model was straight-forward. Development of the models is described further in [1].

III. Architecture of the Diagnostic Experiment

A. EO-1 Satellite Configuration

L2 was integrated with the ASE software architecture and infrastructure, and uploaded to the Wideband Advanced Recorder Processor (WARP) on-board the EO-1 satellite. The WARP is situated within the satellite avionics as shown in Figure 1. Avionics subsystems communicate over a 1773 command-response data bus. The Command and Data Handling (C&DH) processor serves as the 1773 Bus Controller (BC). Other subsystems such as the WARP are Remote Service Nodes (RSN), which communicate under direction of the BC. The C&DH and WARP both contain radiation-hardened Mongoose-5 (M-5) processors with very limited available CPU, around 8 MIPS. Along with hosting the experiment software, the WARP also runs the flight software for data recording and playback. Other systems relevant to the current scope of the experiment are the Advanced Land Imager (ALI) and Hyperion imaging instruments. ALI and Hyperion transfer image data directly to the WARP via RS-422 serial link.

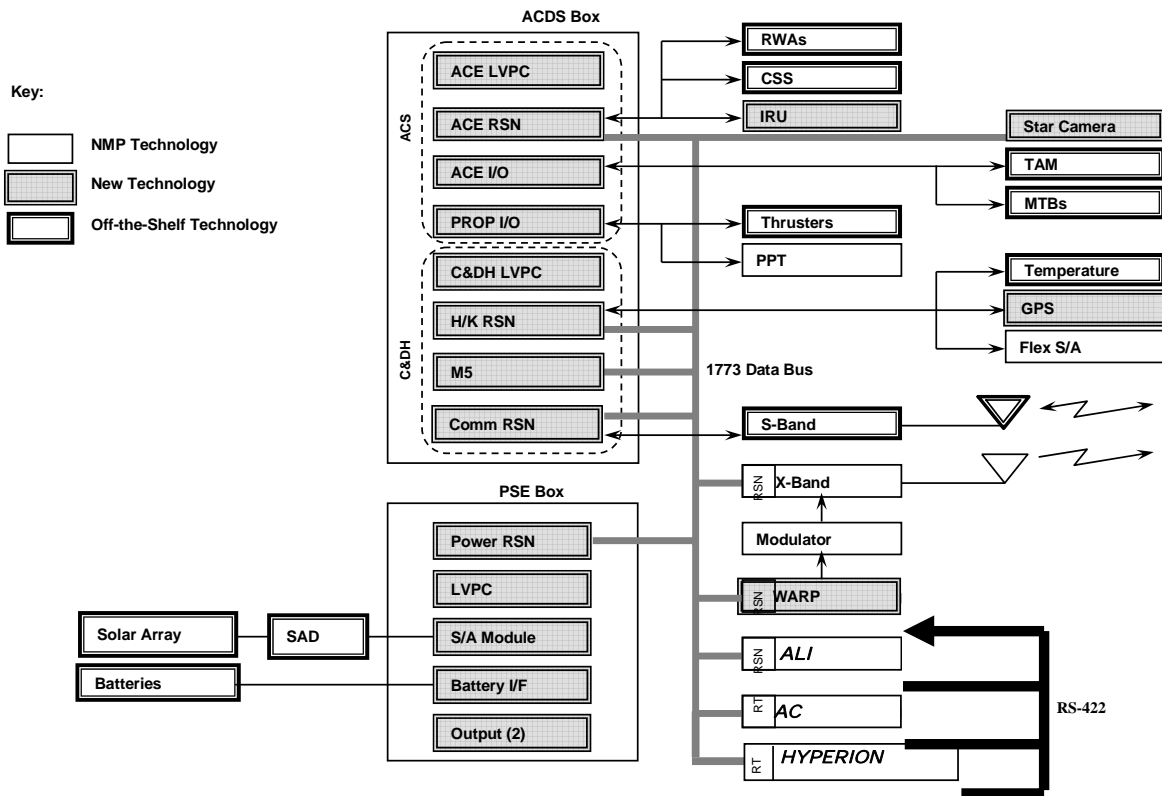


Figure 1: EO-1 Avionics Configuration

L2 runs only during periods in which ASE is in control of the spacecraft as only ASE commands are visible to L2. During normal operations, the C&DH performs commanding of the spacecraft using pre-defined command sequences called Absolute Time Sequence (ATS) or Relative Time Sequence (RTS) loads. When ASE is commanding the spacecraft, the C&DH is silent. The X-band downlink is used for high data rate image transfer to

ground stations, known as playback. Thereafter, the WARP is reformatted to free up memory for further imaging. The S-band is used for low data rate uplink of commands from the Mission Operations Center (MOC) and downlink of spacecraft telemetry for display at the MOC.

B. Software Architecture of the Experiment on the WARP

The experiment architecture and software configuration on the WARP are shown in Figure 2. The diagnostic software has the capability to process spacecraft telemetry and to downlink health status telemetry for monitoring and display at the MOC.

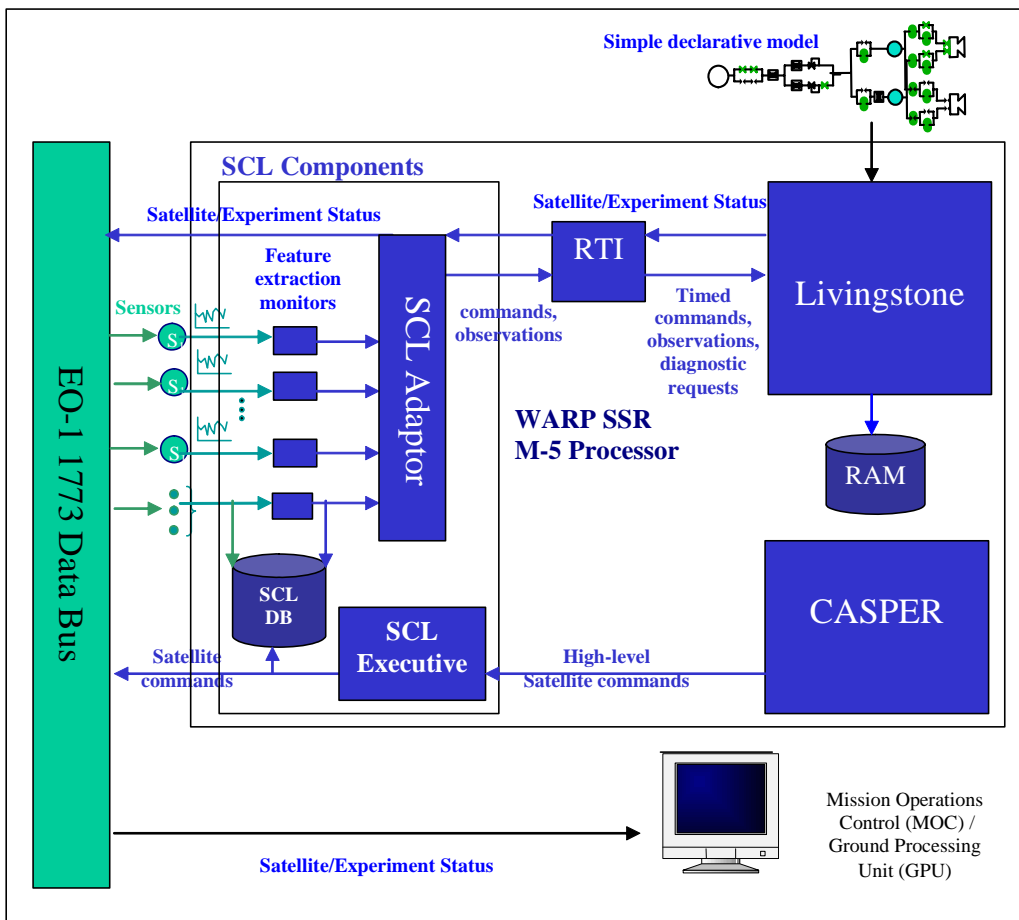


Figure 2: Architecture integrating L2 with ASE on the WARP

The CASPER planner generates high-level plans and sends them to SCL. The SCL Executive executes scripts that send out commands, to execute the plan. The SCL Software Bridge connects applications on the WARP M-5 processor to the 1773 spacecraft data bus for processing incoming telemetry and output of spacecraft commands and ASE/L2 telemetry. The SCL Data Repository stores incoming telemetry data and has database triggers for notification of commands sent and observations received to subscriber processes such as L2.

L2 performs diagnosis using a qualitative model of the target system to predict observations given the commands issued to the system, postulating diagnoses to explain discrepant observations. A diagnosis consists of a group of candidates, their constituent fault modes and the likelihood or rank of each candidate. Each candidate is a set of component fault modes consistent with the current observations, a system-wide hypothesis of what could have gone wrong. The alternate candidates are different hypotheses of faults which can explain the observations, also known as an ambiguity group.

A range of development and integration activities were undertaken to support the experiment. Tasks included development of L2 models of the EO-1 spacecraft and instrumentation, and failure scenario definition, based on knowledge acquired from EO-1 engineers. A Real-Time Interface (RTI) and corresponding modeling methodology were developed to account for communication delays and physical transients in the system. L2, models of the

spacecraft, and the RTI were integrated with SCL and the CASPER planner, first on a PowerPC embedded system and then on EO-1's flight hardware testbed, a pair of M-5 processors with the Virtual Satellite (VSat) simulation system. System engineering of the overall autonomy software included allocation of VxWorks task priorities, SCL software bus bandwidth, CPU and RAM resources. All defined diagnostic scenarios were validated with the integrated software on the flight hardware testbeds, prior to upload. Details of the development, and of the integration and test processes on the flight hardware testbeds, are laid out in more detail in a prior paper [1].

C. L2 Diagnostic Technology

The L2 algorithm and component-connection model are introduced here. An L2 diagnosis system consists of two main parts, a general inference engine and a domain-specific model. When L2 is deployed on different devices or vehicles, the inference engine does not change; only a new model needs to be developed. L2 uses a qualitative representation, propositional logic, to model the target system. The target system may be physical, such as the spacecraft hardware, or logical, such as the spacecraft software. The model is used to predict the states of system components given their initial state, commands which affect the system, and possible mode transitions. If there is a discrepancy between observed and predicted behavior, this generates conflicts in L2's internal belief state. These conflicts are then used to focus the search for component modes (including failure modes), which are consistent with the observed state of the world and the possible mode transitions in the model. This process is known as *conflict-directed best-first search*. The set of component modes, which is found to satisfy the constraints expressed in the model, is termed the mode vector.

An L2 component-connection model describes nominal and failure modes for components in terms of the propositional constraints that must hold in those modes. The connections are the constraints that must hold due to interactions between the components. Transitions between modes of a component are triggered by guard conditions such as commands, similar to a finite state machine representation. The model is qualitative due to the underlying propositional logic representation. Constraints are expressed as discrete variable-value pairs. Any real-valued sensor data must be transformed into qualitative data, "binned" by software called monitors, before being used by L2. For the failure modes, the likelihood of the failure is indicated by a rank, an approximation of the prior probability. The mode vector describes the overall state of the system.

As an example of an L2 model, the Hyperion model is shown in Figure 3 as visualized in the Stanley model development environment.

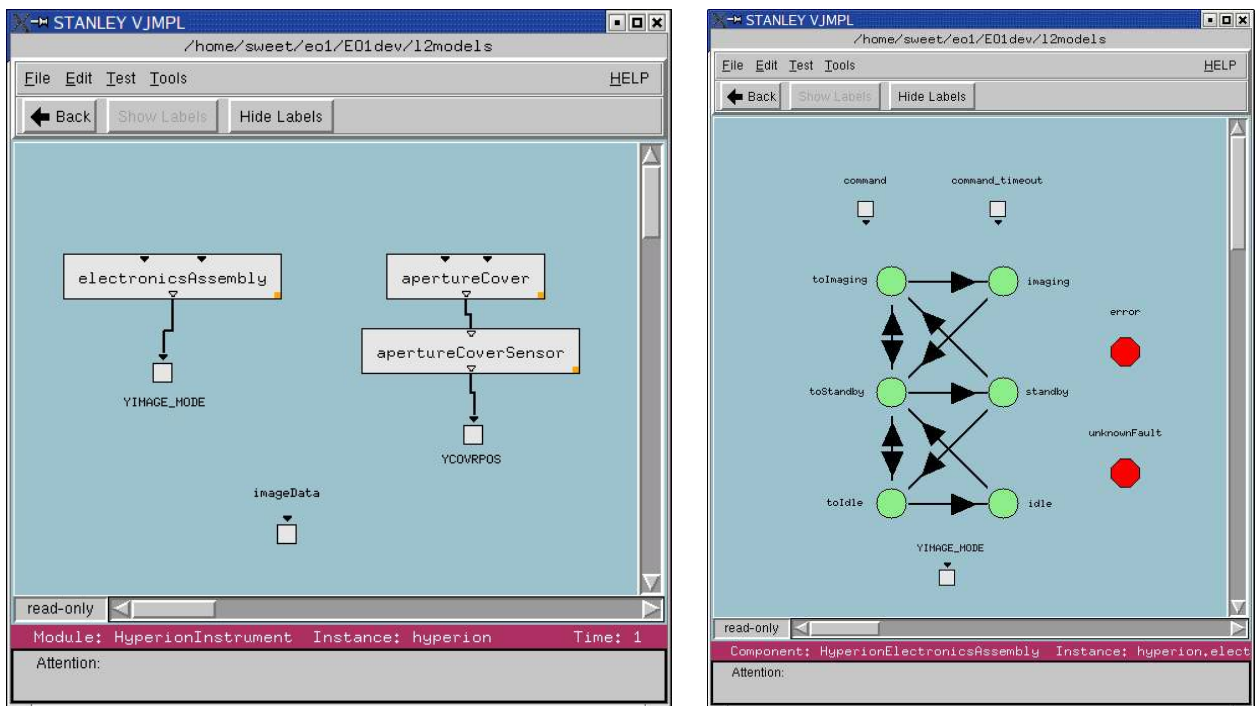


Figure 3: L2 model of the Hyperion subsystem and Hyperion Electronics Assembly Component

There are three components modeled in the Hyperion subsystem; the main aperture cover, which opens to image the earth, the aperture cover sensor, which measures the aperture cover's position, and the electronics assembly, containing the imaging electronics. The *imageData* variable represents what type of image is being taken. It is set based on the modes of the *electronicsAssembly* and the *apertureCover*: NO_IMAGE if the *electronicsAssembly* is disabled, DARK_IMAGE if the electronics are enabled but the *apertureCover* is closed, and EARTH_IMAGE if the electronics are enabled and the aperture cover is open.

The model for the Hyperion electronics assembly component is also shown in Figure 3, as an example of a component model. Here we see the modes and transitions defined for that component. The Hyperion electronics assembly has three nominal modes: *idle*, *standby*, and *imaging*. Usually the instrument is in idle mode. Just before taking an image, the component is commanded into standby mode, sending power to the imagers. The device transitions into imaging mode to start the flow of image data. When the image is complete, the device transitions back to standby, then back to idle if no more images will be taken shortly. The L2 model also includes the transient modes of *toImaging*, *toStandby*, and *toIdle*. The model does not transition directly between the nominal modes; these transient modes are defined to allow any transient effects of the system to die out before the diagnosis. More explanation on the use of transient modes is given in Section D. Finally, there are two fault modes defined: *error* and *unknownFault*. *Error* corresponds to a particular telemetry item reporting the status of the Hyperion electronics, and *unknownFault* is a catch-all mode for all faults not modeled explicitly.

Of course, the sensors on a spacecraft can fail as well as the components. Several sensors with fault modes are included as components in the EO-1 L2 model. As L2 can report multiple hypotheses, often when a failure occurs L2 reports that a component could have failed or a sensor could have failed. Reporting of multiple hypotheses is a key feature to be demonstrated in the experiment.

D. Diagnosis in Real-World Systems

Real-time effects must be accounted for in real-world systems. We have identified two major issues:

- Propagation delays in telemetry communication and software processing links.
- Continuous transients in the physical system.

As an example of the importance of real-time effects, it is believed that the Mars Polar Lander (MPL) spacecraft was destroyed due to premature thrust termination of the descent engine, on detecting dynamic transients in the Hall effect sensor [6, section 7.7]. Shutdown was commanded by the flight software, based on status reported by the touchdown monitor. A requirement to disable Hall sensor input until below 40m altitude had been generated to protect against premature shutdown in the event of transients or failed sensors. However, this requirement did not flow-down to the software requirements. The resulting flight software design did not include a mechanism to protect against transients, permitting the failure to occur.

In hindsight, a cross-subsystem model of the MPL landing phase, with constraints on altitude and landing gear mode, would have generate a conflict for the simultaneous constraints of high altitude and Hall sensors indicating touchdown. A sensor failure would most likely have been diagnosed, either of the altitude sensor or the Hall sensor. With correlating information the controller could have rejected the conflicting information from the Hall sensor and avoided premature engine cutoff.

Propagation delays may be tiny but they are finite. If all observations or commands for a modeled system have not yet been received and L2 performs a diagnosis, a false alarm or false positive may be reported as conflicts arise due to inconsistencies between new and stale data. At best, L2 may report a false negative (missed diagnosis) due to incomplete information. The design of the RTI for EO-1, in conjunction with a transient modeling methodology, account for propagation delays and continuous transients, allowing sufficient time for all information to be received and for transients to settle. EO-1 does exhibit continuous behavior, most notably in the attitude control system, but this is outside the scope of the current model.

1) The Real-Time Interface (RTI)

The RTI is a software module that resides between the SCL Adaptor and L2. The RTI developed for the PITEX project implemented the design decision to withhold effected observations from L2 until the end of the transient period, so that a diagnosis can be made during a transient period without incurring false positives. This involved complex time-segmenting and reasoning to determine when to withhold and when to forward commands and observations to L2. The RTI contained a list of which telemetry observations are affected by which commands; when a command was received, it removed the telemetry observations associated with that command and reinstated them after the transient period was over. Such decisions required the encoding of domain-specific information in the

RTI to know which observations were affected by the transient. This created a dependency between the RTI and the model, a violation of the engine-model separation principle. In addition, when several commands are being executed concurrently and observations are affected by more than one command, deciding when to withhold and reinstate observations becomes burdensome. The implication of this design decision was twofold: unacceptable risk due to overly complex RTI software and the operational impact that any model change would require corresponding updates to the RTI and upload of the ASE/L2 code, a process which takes over a week.

The RTI developed for EO-1 avoids these two pitfalls: its design is simple and contains no model references; commands and observations are forwarded to L2 immediately on notification by the SCL Adaptor. The new RTI works hand-in hand with the transient modeling methodology described below. Diagnoses can be made during a transient period and during concurrent commanding without incurring false positives, due to the modeling of transient modes.

Both the PITEX RTI and the new RTI developed for EO-1 utilize timers to indicate the end of a transient period and transition to steady state mode, whether due to propagation delays or settling of physical transients. Timeouts are presently triggered by an empirically determined constant for each such component mode transition. By necessity, the timeout is an upper bound on the propagation delay and transient response.

2) Transient Modeling Methodology

Previously, L2 models contained only the steady-state modes of a component, and the model instantly transitioned between the steady-state modes, as shown on the left in Figure 4. The modeling methodology for real-time systems, component models includes transient modes between the steady state modes in the component model, as shown on the right. The transient modes are loosely constrained, allowing transient as well as steady state observations to be reported for mode identification without the need to withhold information.

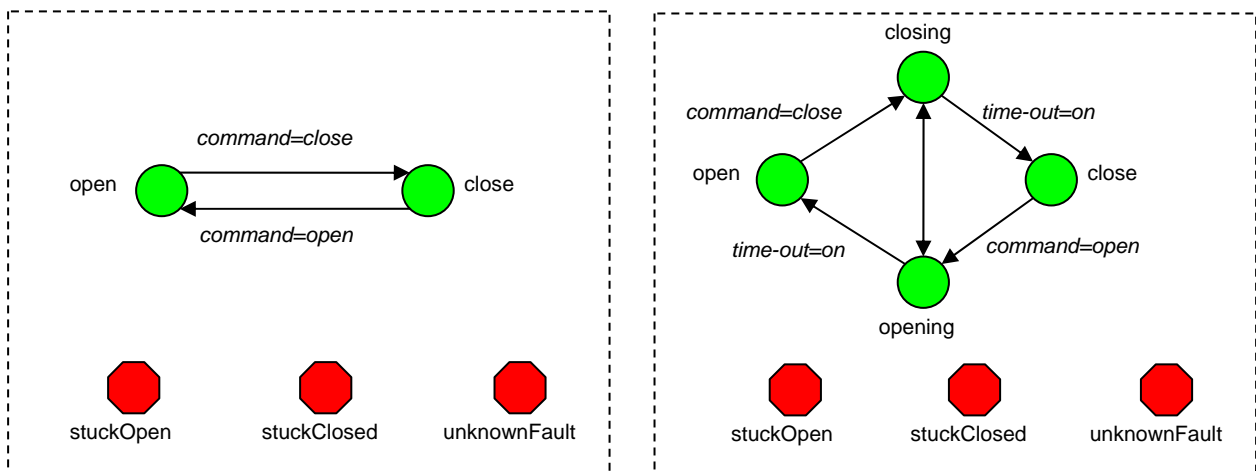


Figure 4: L2 Component Models with and without Transient Modes

The transition modes are inserted between the steady-state modes, e.g. instead of transitioning directly from ‘open’ to ‘closed’, the component transitions from ‘open’ to ‘closing’, then finally to ‘closed’ when we expect the observations to have settled. The transition modes are different from steady-state modes in that they contain no constraints. Thus, L2 has no expectation for telemetry values at this point, the telemetry observations can have any values and L2 will not generate a false alarm. When L2 is signaled that the transient period is over, it transitions into the final steady-state mode, and the constraints of the steady-state mode are imposed. Further, other components which are not affected by the component in transition may be diagnosed as usual. In effect, by including the transient modes, instead of the RTI withholding and reinstating potentially conflicting telemetry observations, L2 removes and reinstates the constraints as the component transitions through transients.

In the case of blind commanding or rapid commanding, the component model facilitates direct transitions between transient modes, as can be seen from the short-cut between ‘opening’ and ‘closing’ modes in the transient model.

The transient period timeout need not be fixed at the upper bound, early timeout and transition to the steady state is possible and desirable to improve diagnostic performance. With monitor support, recognition of the end of the transient period allows early transition to the steady state, diagnosis of a failed transition, and identification and diagnosis of anomalous transients. Monitoring and diagnosis of systems during physical transients is an active area

of research, Hybrid Diagnosis. Continuous systems, such as propulsion systems, exhibit dynamic behavior with hard real-time diagnostic requirements - steady-state diagnosis is no longer sufficient.

IV. Mission Operations for the Experiment

A. Ground Operations

Ground Operations for EO-1 are conducted at the GSFC Mission Operation Center (MOC). The MOC is connected to a network of ground stations which receive telemetry from and issue commands to the satellite. EO-1 is in a polar Sun Synchronous orbit and follows in formation 1 minute behind Landsat-7. The primary ground stations are in Poker Flat, Alaska and Svalbard, Norway and the Tracking and Data Relay Satellite System (TDRSS) can also be used to command and receive lower rate telemetry.

The command and control software in the MOC is called Advanced Spacecraft Integration and System Test (ASIST). Ground commands are uplinked to the satellite through ASIST and telemetry is displayed via Standards-based, Advanced Man-Machine Interface (SAMMI) pages. A screenshot of the SAMMI telemetry page developed for the L2 application is shown in Figure 5.

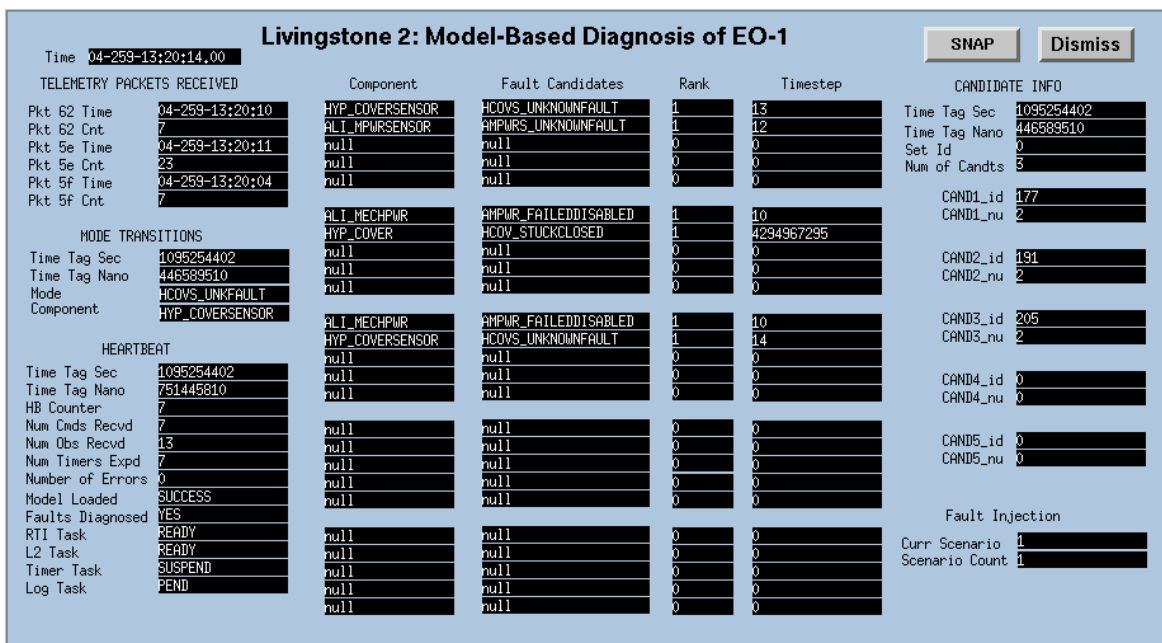


Figure 5: Example of L2 Telemetry Display - the L2 Checkout Procedure

L2 uses three separate telemetry packets: one for diagnostic candidates, one for all mode transitions (both nominal and off-nominal), and one for a task status heartbeat. For instance, in the screenshot, there are three candidates for the L2 checkout procedure. All are double faults: (1) the Hyperion cover position sensor in an unknown fault and the ALI mechanism power sensor in an unknown fault, (2) the ALI mechanism power failed disabled and the Hyperion cover stuck closed, and (3) the ALI mechanism power failed disabled and the Hyperion cover position sensor in an unknown fault. The ranks of these double fault candidates are each two (displayed as Rank value of 1 for each component fault), making this a less likely candidate than a rank one single fault.

In order for spacecraft and ground to process L2 telemetry, the exact data format with byte ordering and sizes in each packet is specified using the Record Definition Language (RDL). All L2 telemetry consists of integer values; this encoding consumes less bandwidth. Labels are used in the RDL specification to map these encoded values to meaningful text.

The diagnostic output is also stored in integer-encoded log files on the WARP. After a test, the log files are compressed and downlinked to the ground, to be decompressed and decoded to human-readable text, for analysis.

To support mission operations, certain commands can be issued by the MOC during a ground contact, using predefined procedures written in the Systems Test and Operations Language (STOL). A suite of STOL procedures has been developed for L2 operations, and documented in the EO-1 L2 Operations Procedures document, to handle:

- Startup of the four L2 VxWorks tasks and the SCL2L2 Adaptor

- Changing of command timeout delays (utilized by the RTI) from the default values in the SCL database.
- Reset of the scenario count, which limits the number of L2 test scenarios which can be run. This is done to conserve ramdisk space.
- Procedures to download L2 log files for analysis, and to remove them from the WARP ramdisk once successfully downloaded.
- Shutdown of the four L2 VxWorks tasks.
- L2 checkout procedures, including a dark calibration procedure which exercises all imaging commands except for the opening and closing of the Hyperion and ALI instrument covers.

Operations procedures may be used for updating the L2 model as well as for SCL database, project and script updates. These require no code changes. All code changes require complete upload of the ASE/L2 build, a significant undertaking of several days.

B. Space Operations

In order to operate the experiment in the blind, the scheduling and execution of on-board tests is automated. L2 testing with any significant fault coverage would proceed very slowly, if commanded solely from the ground. Ground contacts tend to be short, with only about 10 minutes to execute procedures, and time between contacts of about 3 hours. The JPL CASPER planner provides the solution: autonomous in-space operations, which can be conducted out of ground contact. Scenario validation activities are specified in the CASPER goal file, as for any other activity to be planned. In initial tests, a Data Collect Event (DCE) is preceded by an l2-start activity, and succeeded by an l2-terminate activity. This runs L2 only during periods of activity that are in the current model scope. Terminating L2 between fault scenarios is also necessary to clear the previously diagnosed fault, prior to running the next scenario.

It is also possible to run L2 for indefinite periods. In this experiment, we intend to run L2 for extended periods to monitor nominal operations, and hope to discover an actual fault.

V. Results of L2 Validation on EO-1

A. L2 Checkout

In the L2 checkout tests, L2 and SCL execute without CASPER. Commands to the spacecraft are sent up from the MOC, in the form of STOL procedures. For the first test, L2 Checkout, several mode transitions are commanded and monitored and two unrelated faults are injected and diagnosed. The diagnostic telemetry from L2 Checkout as displayed in ASIST is shown in Figure 5. For the second test, a nominal dark calibration was performed. This exercises all commands in the imaging sequence as well as their timeout delays, other than the opening and closing of the instrument covers.

The criteria for success are as follows:

- 1) The diagnosis contains the injected fault as a candidate
- 2) All other candidates in the diagnosis are also possible given the commands and observations

The results for all of the L2 checkout tests are summarized in Table 2. As evident from the table, the L2 checkout tests completed successfully.

Table 2: Results of L2 Checkout Procedures

Checkout Procedure	Diagnostic Candidates	Telemetry Correct?	Logs Correct?
L2 Checkout	ALI mechanism power sensor=unknown fault Hyperion cover position sensor = unknown fault	Y	Y
	ALI mechanism power = failed disabled Hyperion cover = stuck closed		
	ALI mechanism power = failed disabled Hyperion cover position sensor = unknown fault		
L2 Dark Calibration	None. Nominal scenario.	Y	Y

The screenshot in Figure 5 shows the L2 telemetry page for the L2 Checkout Procedure. The relevant diagnostic information is that there are 2 candidates for this fault. Each candidate is a possible explanation of the current

observations. The first candidate contains a single fault, that the ALI's aperture cover is stuck closed. The second candidate contains two faults, that both of the LED sensors have failed. Each candidate is a possibility, according to the observations, but the single-fault candidate is more likely to have occurred than the double fault (as indicated by the lower number in the "Rank" column. Here, the two LED sensors were measuring the position of the aperture cover. Hence, L2's diagnosis is that either the aperture cover is *stuckClosed*, or both of the sensors measuring the cover position have failed. This split of "component failed or sensors failed" is a common result when using L2.

B. Scenarios Demonstrating the Minimum Success Criteria

The results of the scenario testing completed thus far are shown in Table 3. All scenarios tested have completed successfully. No technical problems have occurred.

Table 3: Scenario Validation Test Results

No.	Scenario ID	Sub-system	Fault Injected	Final Diagnosis Candidate(s) and Component Fault(s)	Passed?
1	DCE		None. Nominal scenario.	no component faults	Y
2	Dual		None. Nominal scenario.	no component faults	Y
3	FS01	ALI	Data-Gate Failed Disabled		Untested
4	FS02	ALI	Data-Gate Failed Enabled	data-gate failed enabled data-gate unknown fault	Y
5	FS03	ALI	Mechanism Power Failed Disabled		Untested
6	FS04	ALI	Mechanism Power Failed Enabled		Untested
7	FS05	ALI	Mechanism Power Sensor Failed	mechanism power sensor unknown fault	Y
8	FS06	ALI	Aperture Cover Failed Closed	aperture cover failed closed led08 unknown fault aperture cover stuck transitioning	Y
9	FS07	ALI	Aperture Cover Failed Open		Untested
10	FS08	ALI	Aperture Cover Failed Intermediate		Untested
11	FS09	ALI	LED 09 Failed	LED 09 unknown fault	Y
12	FS10	ALI	LED 08 Failed		Untested
13	FS20	HSI	Aperture Cover Failed Open	aperture cover stuck open aperture cover sensor unknown fault	Y
14	FS21	HSI	Aperture Cover Failed Closed		Untested
15	FS23	HSI	Electronics Error		Untested
16	FS24	HSI	Aperture Cover Sensor Failed		Untested
17	FS35	WARP	Failed To Record	software unknown fault	Y

In this section, scenarios which meet each Minimum Success Criteria requirement are explained

1) Spacecraft Hardware in the Loop

Nominal and dual nominal scenarios demonstrate nominal operations. All other scenarios demonstrate fault diagnosis. All scenarios are executed on-board with spacecraft hardware and software in the loop.

2) Multiple Hypotheses

Scenario number 6, the ALI mechanism power failed enabled, meets this criterion. More than one possible explanation for the failure is given. In this scenario, the telemetry value reporting the mechanism power remains "enabled" at a time when the mechanism power should be "disabled". As a result, L2 reports two possible candidates: that the mechanism power component has failed in the "enabled" mode, or the sensor measuring the mechanism power has failed. These two possibilities are indistinguishable according to the observations, and both remain throughout the scenario.

3) Multiple Hypotheses with Backtracking

Scenario number 11, the "open LED sensor" fault, meets this criterion. The likely hypotheses are revised based on updated observations received on the state of the spacecraft. Here, the "open LED sensor", which reports the state of the ALI aperture cover as "fully open" or "not fully open", fails. The fault is injected before the ALI aperture cover is opened, and the sensor reports "not fully open" for the entire scenario. The fault is detected after attempting to open the ALI aperture cover, and the two candidates from L2's diagnosis are (1) that the LED sensor has failed, or (2) that the ALI aperture cover has failed in the intermediate position. Again, both are possibilities given the observations. However, later in the scenario the ALI aperture cover is commanded to close, and we get an update from the "closed LED sensor" reporting that the ALI aperture cover is fully closed. This couldn't have happened if the ALI cover was in fact failed in the intermediate position, and as a result L2 rules out this possibility. The only candidate that persists to the end of the scenario is the sensor fault.

4) Diagnosis During Transients

Diagnosis of a failure shall not be delayed by concurrent commanding of the spacecraft:

All scenarios satisfy this criterion. During a data collection, several components are commanded at once. The separate commands given the separate components are diagnosed independently, based on the best information available at the time of diagnosis.

5) Separation of Code and Model

The RTI developed for L2 on EO-1 is domain-independent and does not have this limitation. All scenarios demonstrate this.

The last two feature requirements, which are not MSC requirements, have not yet been validated:

- 1) Coverage of diagnostic scenarios_- several of the defined seventeen scenarios are not yet tested
- 2) Long-term space operations

In the onboard tests, all eight executed scenarios have completed successfully. Nine of the seventeen scenarios remain to be tested. We expect scenario FS01 to fail, if onboard timing latencies are similar to those experienced on the flight hardware testbed. During ground testing, it was found that the ALI data gate was commanded enabled, but commanded disabled again before the "enabled" telemetry was received. Therefore, L2 saw the same final disabled mode and did not have timely information to detect that the intermediate mode transition had failed. L2 assumes no faults exist until evidence to the contrary is received; in this case, that assumption resulted in a missed diagnosis or false negative for FS01 during ground testing.

A minor difference between the on-board test results and L2 ground unit testing exists, with FS06 reporting a different double-fault candidate. These candidates are both equally likely. Because of CPU restrictions, L2 is restricted from exhausting all possible candidates. Here, it simply found this double-fault candidate first. If L2 was not limited by CPU, it would have returned both double-fault candidates (for a total of 3 candidates) in both the unit test and the on-board test.

VI. Conclusion

Over the past year, the project has gone from initiation to deployment on-board a spacecraft. Models of the satellite were developed from scratch and diagnostic scenarios validated on a series of testbeds of increasing fidelity. A new Real-Time Interface and transient modeling methodology were employed to enable the software to run on a real-world system, with reduced complexity and complete independence of the RTI from the model specification. The project team learned about the satellite, about operations procedures, and how to coax delicate hardware and firmware systems into a working state. In September, L2 checkout on-board EO-1 was successfully completed and diagnostic scenario validation commenced. Tests are expected to continue until December.

This experiment was undertaken primarily to mature L2 in a flight environment and ready it for technology transfer to industry. This has been accomplished. TRL 7, defined as 'System prototype demonstration in a space environment', has been reached. The L2 source code has been made open source, available at [8].

Continued effort in the area of model-based diagnosis and system health management is essential to further develop this critical technology for infusion into NASA's missions. Further important work remains that we would

like to address, such as enacting recovery once a diagnosis is made by closing the loop with the CASPER planner, with performance improvements to meet hard real-time requirements. The L2 models could be grown to cover additional subsystems such as power and attitude control, to self-diagnose faults within the autonomy software, and to diagnose known faults that have occurred on-board the spacecraft.

Acknowledgements

The work of the L2/Skunkworks team on the development and maintenance of L2 and Stanley must be recognized. Also, thanks to the PITEX team and the X-37 IVHM team for providing realistic applications to exercise L2, and for maturing the technology with the VxWorks real-time operating system and the PowerPC testbeds. Without this work to lay the foundation, the experiment would not have been possible.

References

1. S. Hayden, A. Sweet and S. Christa, "Livingstone Model-Based Diagnosis of Earth Observing One". *Proceedings of AIAA 1st Intelligent Systems Conference*, Chicago, 2004.
2. B. Williams and P. Nayak, "A Model-based Approach to Reactive Self-Configuring Systems". *Proceedings of Thirteenth National Conference on Artificial Intelligence (AAAI'96)*, Portland, Oregon, 1996.
3. N. Muscettola, P. Nayak, B. Pell and B. Williams, "Remote Agent: To Boldly Go Where No AI System Has Gone Before". *Artificial Intelligence*, Vol 100, Best of IJCAI 97
4. S. Chien, R. Sherwood, D. Tran, R. Castano, et al., "Autonomous Science on the EO-1 Mission". *Proceedings of the Seventh International Symposium on Artificial Intelligence, Robotics and Automation in Space*, Nara, Japan, 2003.
5. J. Kurien and P. Nayak. "Back to the future for consistency-based trajectory tracking". *Proceedings of the 7th National Conference on Artificial Intelligence (AAAI'2000)*, 2000.
6. JPL Special Review Board, "Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions".
7. C. Meyer, H. Cannon, E. Balaban, C. Fulton, B. Maul, A. Chicatelli, A. Bajwa, and E. Wong, "Propulsion IVHM Technology Experiment Overview". *Proceedings of IEEE Aerospace Conference, Big Sky, Montana, 2003*.
8. Livingstone 2 Open Source at <http://opensource.arc.nasa.gov/>

Biographies

Sandra Hayden is principal investigator and project manager of the Livingstone on EO-1 (LEO-1) Infusion Experiment, and a group manager for QSS Group Inc. Founded the QSS Software Process Improvement Network and lead CMMi pre-appraisals of selected QSS tasks. Architected and developed Integrated Vehicle Health Management (IVHM) for reusable launch vehicles (RLVs), under the Space Launch Initiative program. Ames Project Manager, under the Next Generation Launch Technology program, for the PITEX project, an IVHM application for the X-34 RLV Main Propulsion System. Her interests include model-based diagnosis and flight-critical systems. Prior experience is on software engineering for large, mission and safety-critical Ada software systems such as the Canadian Automated Air Traffic System (CAATS) and a 1553 System Data Bus for a submarine; and operations software for undersea diamond mining. Ms Hayden received an MS in Computer Science from Simon Fraser University, Vancouver, Canada.

Adam Sweet is a research engineer in the Computational Sciences Division at NASA Ames Research Center, under contract to QSS Group Inc. He graduated with an MS in Mechanical Engineering from UC Berkeley in 1999, and has since worked at Ames modeling and simulating physical systems. His focus has been in robotics, hybrid system simulation, and model-based diagnosis.

Scott Christa is a systems engineer at NASA Ames Research Center under contract to AerospaceComputing, Inc. He graduated with a Bachelor of Science in Aeronautics from San Jose State University and holds an Airline Transport Pilot (ATP) and flight instructor certificates, although he doesn't fly for a living. He specializes in programming embedded system using anything from high-level languages down to assembler code.

Daniel Tran is a member of the technical staff in the Artificial Intelligence Group at the Jet Propulsion Laboratory, California Institute of Technology, where he is working on developing automated planning and scheduling systems for onboard spacecraft commanding. Daniel attended the University of Washington and received a B.S. in Computer Engineering, graduating with honors. He is currently the software lead for the Autonomous Sciencecraft Experiment flying onboard the Earth Observing-1 satellite.

Seth Shulman is the EO-1 Flight Operations Team (FOT) Technical Lead at NASA GSFC under contract to Honeywell Technology Solutions Inc. (HTSI). He graduated with a BS in Electrical Engineering from the University of Maryland and has since worked at GSFC supporting both the Flight Dynamics Facility (FDF) and EO-1 Mission Operations Center (MOC). His focus has been in attitude determination & control, orbit determination, and maneuver planning.