AIAA Infotech@Aerospace 2010
20 - 22 April 2010, Atlanta, Georgia

AIAA 2010-3547

Lessons learned on implementing Fault Detection, Isolation, and Recovery (FDIR) in a Ground Launch Environment

Bob Ferrell[1] , Mark Lewis[2] and Jose Perotti[3]
NASA, Kennedy Space Center, FL, 32899

Rebecca Oostdyk[4] and Jesse Goerz[5]
ASRC Aerospace, Kennedy Space Center, FL, 32899

and

Barbara Brown[6]
NASA, Ames Research Center @ KSC, Kennedy Space Center, FL, 32899

This paper's main purpose is to detail issues and lessons learned regarding designing, integrating, and implementing Fault Detection Isolation and Recovery (FDIR) for Constellation Exploration Program (CxP) Ground Operations at Kennedy Space Center (KSC).

I. Introduction

Part of the overall implementation of National Aeronautics and Space Administration's (NASA's) Constellation Exploration Program (CxP), Fault Detection Isolation and Recovery (FDIR) is being implemented in three main components of the program (Ares, Orion, and Ground Operations/Processing). While not initially part of the design baseline for the CxP Ground Operations, NASA felt that FDIR is important enough to develop, that NASA's Exploration Systems Mission Directorate's (ESMD's) Exploration Technology Development Program (ETDP) initiated a task for it under their Integrated System Health Management (ISHM) research area. This task, referred to as the FDIR project, is a multi-year multi-center effort. The primary purpose of the FDIR project is to develop a prototype and pathway upon which Fault Detection and Isolation (FDI) may be transitioned into the Ground Operations baseline. While not discussed in this paper, additional details of how this FDIR project fits into the overall NASA structure is available in Ref. 1.

II. Fault Isolation Tool

Currently, Qualtech Systems Inc (QSI) Commercial off the Shelf (COTS) software products Testability Engineering and Maintenance System (TEAMS) Designer and TEAMS RDS/RT are being utilized in the implementation of FDI within the FDIR project. The TEAMS Designer COTS software product is being utilized to model the system with Functional Fault Models (FFMs). A limited set of systems in Ground Operations are being modeled by the FDIR project, and the entire Ares Launch Vehicle is being modeled under the Functional Fault Analysis (FFA) project at Marshall Space Flight Center (MSFC). Integration of the Ares FFMs and the Ground Processing FFMs is being done under the FDIR project also utilizing the TEAMS Designer COTS software product.

[1] Lead Electronics Engineer, Advanced Systems Branch, Mailstop: NE-E9
[2] Electronics Engineer, Advanced Systems Branch, Mailstop: NE-E9
[3] Chief, Advanced Systems Branch, Mailstop: NE-E9
[4] Electrical Engineer, Advanced Electronics and Technology Development, Mailstop: ASRC-25
[5] Computer Engineer, Advanced Electronics and Technology Development, Mailstop: ASRC-39
[6] ISHM-FDIR Task Lead, NASA-Ames Resident Office, Mailstop ARC

The TEAMS tool was chosen for this ETDP FDIR project based on commonality with Orion and the Ares Launch vehicle. Both of these CxP elements chose TEAMS based on a CxP initial concept development report[2] entitled "Diagnostic Technology Evaluation Report for On-Board Crew Launch Vehicle" that was performed by NASA Ames Research Center (ARC). This report evaluated the state of art practice in embedded fault detection and diagnosis technologies for Crew Launch Vehicle (CLV) requirements development.

After creating the FFMs in the TEAMS Designer COTS software, the modeler can utilize various reporting and analysis tools within the TEAMS Designer COTS software to verify the functionality of the model. The TEAMS Designer COTS software contains a TEAMS-RT Analysis Tool and Design for Testability (DFT) feedback which gives the modeler the opportunity to inject failure modes and failed test results into the model to test its functionality and do limited validation.

Ultimately, the TEAMS Designer COTS software outputs a Dependency-Matrix (D-Matrix) that maps failure effect propagation paths from failure modes identified in the FFMs to observable points, or test points, in the system. The TEAMS RDS/RT COTS software then uses the D-Matrix in real-time to determine which modules in the model are bad, suspect, or unknown, based on test results that are passed to it. In order for the TEAMS RDS/RT COTS software product to function correctly, custom software must be developed for interfacing to the CxP Ground Operations Launch Control System (LCS) and implementation of tests for resultant passage to the TEAMS RDS/RT COTS software.

III. Anomaly Detection Tool

Currently, the ARC developed Inductive Monitoring System (IMS) has been chosen as the anomaly detection tool. IMS uses a data-driven machine learning technique that automatically extracts system parameter relationships and interactions from archived nominal system data producing a monitoring capability quickly for almost any system for which archived nominal data is available. IMS does not require knowledge engineers or modelers to capture precise details of system operation. It only requires archived nominal data. Lacking that, IMS can learn the relationship based on the parameter values from a high-fidelity simulator, with the accuracy of the simulator determining the accuracy of learned monitoring capability

IMS has two phases: training and monitoring. Training is done off-line and monitoring can be done on-line (real-time monitoring) or off-line (post-flight analysis). The goal during the training phase is to learn how the system normally behaves. The input to the learning algorithm is a data set representing nominal system operation.

The goal of the monitoring step is to determine if the system is behaving differently than during operations when training data was collected. The monitoring step extracts the relevant parameters from the incoming real-time system data, normalizes and weights them, as was done during training, and then compares the incoming vector to the clusters generated during training. It outputs similarity scores, the distance between the vector and the nearest cluster for the vector as a whole (a composite score) and a distance from the nearest cluster for each parameter separately. These distances represent the deviation of the current operation from nominal operations as defined by the training set – a measure of how "out of family" the behavior is. More detailed info on IMS is available in Ref. 3.

IV. Issues & Lessons Learned

The following section outlines issues that based are on our experience. We feel that should be considered, were encountered, or were significant lessons learned so far, during development of FDIR for this project.

A. Architecture

FDIR is intended and designed to be integrated with Ground Operations to automate fault detection and isolation during maintenance and checkout, as well as launch countdown activities of ground and launch vehicle systems. In addition, the FDIR architecture will support the integration of several ISHM capabilities or tools. This not only entails the design of an appropriate internal architecture, but also an external architecture/interface that is capable of integrating into the desired operational command and control system.

The FDIR project team evaluated numerous architectural alternatives in determining what they feel is the optimum architectural solution for FDIR's internal architecture. While sometimes design is done for designs sake, one of our overriding concerns was for future sustaining engineering. This is with not only within the internal FDIR architecture but also in how it is integrated with LCS. For more details on the architecture see Ref. 1

B. Tool Selection

While this project's tools were basically pre-determined, a number of observations have been experienced that should help in evaluating and/or selection of additional tools. The first of these is to understand how a tool works during design time versus run time. On some tools, features are not consistent between the development tool and its corresponding run-time environment. In addition, there has been significant progress in COTS FDIR/ISHM tools during the past decade. Some of this progress has involved complete recoding of tools and/or new developments. While there can be ITAR/Export issues, some of these tools are from foreign corporations. Considering the impact of an integrated FDIR capability on a command and control system, this evaluation and selection of tools should not be underestimated. However, having an architecture that supports multiple tools will ease the impact of adding or changing tools.

C. Model Development

For the TEAMS tool suite that was chosen, FFMs were developed for a target system. Due to their complexity and failure history, cryogenic systems were targeted for initial development in the FDIR prototype. In particular, we have designated the initial subsystems for prototype integrations to be the Ground Liquid Hydrogen (LH2) subsystem and the Ares Upper Stage LH2 Main Propulsion Subsystem (MPS). Having an architecture that supports distributed models can be beneficial; however one needs to ensure that modeling conventions and model Interface Control Documents (ICDs) are done at an early stage of development. This will not only help ensure a smooth integration of the models but will also be beneficial when expanding the scope of the models. An example of this is provided in Ref. 4. Finally, having a standardized approach to model development will ensure that models are similar in nature when developed by different modelers.

Additional details of the functional fault modeling approach that we utilized, and satisfies the TEAMS toolset, is presented in Ref. 5 at this conference.

D. Simulation Capability for Testing

Once the FFMs have been developed and transitioned to run-time, testing must occur with some type of real-time data. For this project's purpose, since CxP elements are in development, data and/or simulations were not available. It is for this reason, that we had chosen elements of Shuttle's LH2 GSE to model. By doing this we were able to retrieve data for actual shuttle operations and with slight modifications create faults for playback to FDIR. While not actual CxP data, this process reduces the risk associated with not having that data or simulations of it available.

E. Integration

The integration of FDIR to the LCS command and control system was handled in Section A, Architecture. Model integration was handled in Section C, Model Development. This section is primarily concerned with the FDIR internal integration after data acquisition. This requires custom software that interfaces CxP LCS and implementation of tests for resultant passage to the TEAMS RDS/RT COTS software and is generically referred to as "wrapper code". This wrapper code must be tailored to the telemetry, TEAMS model, and GUI with which it interfaces, but this results in code that is application specific and requires re-work each time the telemetry, model or GUI changes. Previous wrapper code applications required heavy involvement of TEAMS modeler(s) in the development process which necessitated a special skill mix to get the TEAMS model operating in real-time. A WrapperD application was developed with the intent of isolating the interface functions from one another so they can easily be tailored for other environments and mitigating the dependency of the WrapperD code on the intricacies of the TEAMS model to limit the modeler's involvement in wrapper code development.

In particular, the WrapperD application performs the tasks required for TEAMS RDS/RT COTS software to use the D-Matrix through the TEAMS APIs. These tasks include:

1) Parsing the incoming telemetry stream,
2) Performing logical tests based on the current telemetry data,
3) Passing the results of the logical tests to TEAMS RDS through the TEAMS API,
4) Requesting the "bad", "suspect", and "unknown" diagnoses from TEAMS RDS through the TEAMS API, and
5) Communicating information that is required by a Graphical User Interface (GUI) to display the telemetry, test results, and diagnosis to the user.

Each of these five tasks is as independent as possible. This is done to facilitate re-use of the code in other real-time systems. In addition, some of the code in the WrapperD application is auto-generated to make it re-usable regardless of the content of the TEAMS model.

F. Performance

It is imperative that preliminary performance testing be done as early in the development process as possible. The results of this testing can impact not only tool selection but also architecture decisions. If at all possible, the breaking points of implemented software should be found in order to obtain worst case scenarios. These worst case scenarios can help decide if overall real-time performance requirements can be met. Additional performance details, that we obtained of the TEAMS real-time software is available in Ref. 6.

V. Future Stuff

While we did limited playback of shuttle data for testing, we are planning on developing a low fidelity simulation capability for CxP Ground Ops LH2. This is done in order to do higher fidelity testing and risk reduction for continued development of FDIR. Of course, all of this work is dependent on what direction NASA will take in the next decade of space exploration. The prototype demonstrations that have we have held so far have been hugely successful and highly regarded. We can only hope that this type of activity will continue in the future.

VI. Conclusion

While some of the work performed on this project is highly dependent on the LCS command and control system. A significant amount of the design is internal to the FDIR project and not dependent on outside entities. The following are the major technical issues and lessons learned from the FDIR work that we have performed so far:

Internal Architecture – Should support multiple tools and the addition of tools throughout the lifecycle of the product.

External Architecture – While the interfaces are customized to a specific system for data acquisition and output, remember that sustaining engineering is a significant cost during the lifecycle of a command and control system with integrated FDIR.

Tool Selection – Understand the difference in a tool's design/development features and its run time features and plan accordingly during development. Assess current tools and available tools. Look not only commercially, but also internal, to your organization, at industry, at academia and government institutions.

Modeling – Choice of an initial target system that demonstrates benefits is paramount for continued funding. Define conventions and ICDs early to enhance the capability of adding model scope. Develop a standardized process, as much as possible, for model development.

Simulation/Testing – There should be a plan in place to provide for testing with some type of simulation or playback. This data should resemble the ultimate operational conditions as close as possible in order to provide risk reduction for the overall project.

Internal Integration – Utilize reusable or auto-generated code wherever possible for internal FDIR integration wherever possible. This will help reduce lifecycle costs.

Performance – Preliminary performance testing should be performed as early as possible to ensure that real-time performance requirements can be met.

Acknowledgments

We would like to thank NASA's Exploration Technology Development Program for their past and future funding of the Fault Detection, Isolation and Recovery project to develop and mature fault isolation technologies for future space missions.

References

[1]Ferrell, B., Lewis, M., Perotti, J., Oostdyk, R., Spirkovska, L., Hall, D. and Brown, B., "Usage of Fault Detection Isolation & Recovery in Constellation Launch Operations," Proceedings of the AIAA SpaceOps 2010 Conference, AIAA, Huntsville, AL, 2010. Submitted for publication.

[2]Hayden, S., Oza, N., Mah, R., Mackey, R., Narasimhan, S., Karsai, G., Poll, S., Deb, S., and Shirley, M., "Diagnostic Technology Evaluation Report for On-Board Crew Launch Vehicle," NASA/TM-2006-214552, September 2006.

[3]Spirkovska, L., Iverson, D.L., Hall, D.R., Taylor, W.M., Patterson-Hine, A., Brown, B.L., Ferrell, B.A., and Waterman, R.D., "Anomaly Detection for Next-Generation Space Launch Ground Operations," *Proceedings of the AIAA SpaceOps 2010 Conference*, AIAA, Huntsville, AL, 2010. Submitted for publication.

[4]Ferrell, B., Brown, B., Lewis, M., Oostdyk, R., and Perotti, J., "Functional Fault Modeling Conventions and Practices for Real-Time Fault Isolation," *Proceedings of the AIAA SpaceOps 2010 Conference*, AIAA, Huntsville, AL, 2010. Submitted for publication.

[5]Ferrell, B., Brown, B., Lewis, M., Oostdyk, R., and Perotti, J., "Functional Fault Modeling of a Cryogenic System for Real-Time Fault Detection and Isolation," *Proceedings of the AIAA Infotech@Aerospace 2010 Conference*, AIAA, Atlanta, GA, 2010.

[6]Ferrell, B. and Oostdyk, R., "Modeling and Performance Considerations for Automated Fault Isolation in Complex Systems," *Aerospace Conference Proceedings*, IEEE, Washington, DC, 2010, 978-1-4244-3888