Verification of Functional Fault Models and the Use of Resource Efficient Verification Tools

Rachael Bis[1]
N&R Engineering, Cleveland, Ohio, 44130

and

William Maul[2]
Vantage Partners, LLC, Brook Park, Ohio, 44142

Functional fault models (FFMs) are a directed graph representation of the failure effect propagation paths within a system's physical architecture and are used to support development and real-time diagnostics of complex systems. Verification of these models is required to confirm that the FFMs are correctly built and accurately represent the underlying physical system. However, a manual, comprehensive verification process applied to the FFMs was found to be error prone due to the intensive and customized process necessary to verify each individual component model and to require a burdensome level of resources. To address this problem, automated verification tools have been developed and utilized to mitigate these key pitfalls. This paper discusses the verification of the FFMs and presents the tools that were developed to make the verification process more efficient and effective.

Nomenclature

AGSM	=	Advanced Ground Systems Maintenance
COTS	=	Commercial-Off-The-Shelf
D-Matrix	=	Diagnostic Matrix
ETA	=	Extended Testability Analysis
FFM	=	Functional Fault Model
FM	=	Failure Mode
FMEA	=	Failure Mode and Effects Analysis
GRC	=	Glenn Research Center
TEAMS	=	Testability Engineering And Maintenance System
TMCP	=	Testability Engineering and Maintenance System (TEAMS) Modeling Conventions and Practices
TP	=	Test Point
SME	=	Subject Matter Expert
VERA	=	VERification Analysis
VV&A	=	Verification, Validation and Accreditation

I. Introduction

FUNCTIONAL fault models (FFMs) are used to support the development and real-time diagnostics of complex systems through the use of directed graph representation of the failure effect propagation paths within a system's physical architecture. The verification of these models is required in order to confirm that the FFMs are correctly built and accurately represent the underlying physical system. The purpose of this paper is to discuss the verification of the FFMs and present the tools which have been developed to make the verification process more efficient and effective.

FFMs are being used to support system design by verifying diagnostic requirements and to automate the isolation of faults during the operation of advanced ground and space systems. A qualitative approach to functional fault modeling was selected in order to support early system design and development and still provide a real-time diagnostic

[1] Controls Engineer, GRC-LCC, 21000 Brookpark Rd, MS: 77-1, Cleveland, OH 44135.
[2] Aerospace Engineer, GRC-LCC, 3000 Aerospace Parkway, MS: VPL-3, Brook Park, OH 44142.

engine in the field. The FFM of a system should be developed as a component-based model that mimics system engineering products, such as FMEAs (Failure Mode and Effects Analysis), enables varying levels of fidelity across the system and allows for sectional sub-model development that can be integrated as required.

The failure effect propagation paths within the FFMs connect failure modes associated with system components, to all of the components which the failure may affect and the sensors which can detect the effects. This propagation occurs within a hierarchical framework modeled on the architecture of a system, which is usually based on a system schematic. The FFMs can be used to analyze the detection coverage and failure mode isolation capability of the system's suite of sensor measurements. For the purposes of this paper, the FFMs being considered were developed using the COTS (Commercial Off The Shelf) software package: Testability Engineering And Maintenance System (TEAMS) Designer from QualTech Systems Inc[1].

A representation of an FFM modeled in TEAMS is shown in Fig. 1. The hierarchical framework of TEAMS Designer-based FFMs includes failure modes associated with system components, failure propagation paths, mappers, and the sensors and tests used to identify the faults in a system. For example, in Fig. 1, the top most failure mode could produce the failure effect of a hydraulic fluid leak (i.e. a possible way the component could fail is by developing a leak in the hydraulic system). This failure effect is propagated along the link (black arrow) to the subsequent mapper, where the failure effect is transformed from a leak to a low hydraulic fluid flow failure effect. The links/arc are colored to represent the nature of the failure effects that they pass (e.g. green for hydraulic fluid related failure effects, black-dashed for power related failure effects, etc.). The low hydraulic fluid flow failure effect then exits the component and then, presumably, enters another component or part of the higher level system via the component's output port. The failure effect could later be detected by a test, which represents a sensor or observation point in the physical system.

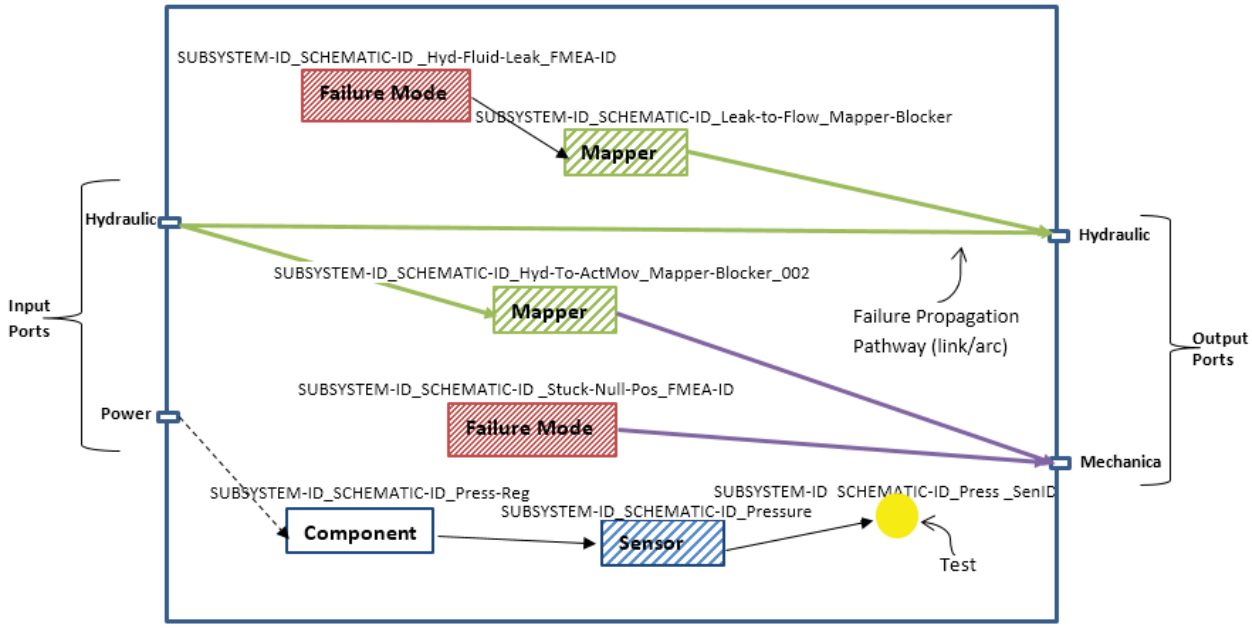


Figure 1. A sample generic functional fault model developed using TEAMS Designer.

For the graphical FFM representation, "black boxes" are used to produce, propagate, transform, and detect failure effects instead of numerical simulations of physical effects. This representation allows system developers to determine if a system has full failure detection coverage before the specifications of individual components have been fully defined, thus allowing detection coverage information to be used earlier in the devlopment cycle. However, as will be addressed later in this paper, these FFMs must still be fully verified but the verification options are restricted due to the graphical and qulatitave nature of the FFMs.

Verification of these models is driven by NASA-STD-7009[2] "Standards for Models and Simulation," which specifies the verification requirements necessary to determine the credibility of a model as part of the model accreditation process. The verification portion of the model accreditation process described in this paper was originally developed to support the Advanced Ground Systems Maintenance (AGSM) Project at NASA's Kennedy Space Flight Center. However, a manual, comprehensive verification process applied to the FFMs was found to be error prone and

burdensome due to the individualistic nature of the process and the high level of resources required to implement it. To address these problems, automated verification tools have been developed and utilized to mitigate these key pitfalls.

The intent of this paper is to describe the verification process that has been developed for FFMs and to present the tools with which this process has been made more efficient and accurate. Section II of this paper discusses the different methods of verification that are being used with the FFMs, including static and dynamic verification. Section III opens with a brief review of the verification literature that motivated the development of two tools which are used to facilitate the verification. These tools, also described in Sec. III, are the VERification Analysis (VERA) tool, which partially automates model implementation verification, and the Extended Testability Analysis (ETA) tool, which performs unit testing. And finally, Sec. IV provides some final remarks on the work presented in this paper.

II. Verification of FFMs

Verification of simulation models is a developing field with no universally adopted method or set of verification procedures. However, significant research has been performed on developing specific verification techniques and determining the scenarios for which they are most appropriate and valuable. Model verification, which is commonly defined as "ensuring that the computer program of the computerized model and its implementation are correct,"[3] can be generally broken down into static and dynamic verification tests, where the static tests involve the analysis of the program code (or model) and the dynamic tests involve the execution of the program code (or model) under different conditions.[4] Both forms of verification are necessary to verify that the model or simulation is free of errors.

For the purposes of this paper, a fully verified model must affirmatively answer the question put forth by NASA-STD-7009, "Is the model built correctly?" When this standard is applied to FFMs[5], the objective of verification is to directly measure the quality of an FFM in order to determine if the model was built correctly from the perspective of its intended use. Comprehensive verification to answer this question requires three different forms of testing and review: verification of correct model implementation, which is a static form of verification, and unit and regression testing, which are dynamic forms.

The FFMs being verified in this paper have been created in a graphical environment, as shown in Fig. 1, instead of a more traditional, text-based environment. The graphical form of functional abstraction used by the TEAMS-Designer software allows for an intuitive development of system models based on the system's schematics and other data. However, like text based code, the implementation and operation of the FFM must still be verified to confirm that the FFM adheres to the established conventions and performs in the expected manner. While this verification would typically be performed via numerical and syntactical analysis for a quantitative, text-based mode, the qualitative and graphic nature of FFMs require that unit testing and model implementation verification techniques be used instead.

A. Static Verification of Model Implementation

Verification of the model implementation of FFMs consists of inspection of the model to ensure that it is complete and includes all pertinent systems and scenarios, conforms to current modeling conventions and practices as defined in Testability Engineering and Maintenance System *(TEAMS)* Modeling Conventions and Practices[6] (TMCP), and accurately reflects the reference material and system schematics used for its development. The FFM modeling conventions and practices are based, in part, on previously developed best practices for fault isolation[7]; however, no prior attempt appears to have been made to formally verify the adherence of FFMs to a specific set of conventions.

The verification method presented in this paper follows the method set forth in the Functional Fault Modeling and Fault Isolation Verification, Validation and Accreditation Plan[5] (VV&A Plan) that was written to support the AGSM Project. In this plan, errors in the model implementation are divided into categories, based on the part of the model that they effect, and scores are assigned to each error based on that error's severity. These errors and the resulting scores are divided into three different groups: technical errors, practice/convention errors and cosmetic errors. Each of these types of errors are scored independently so that both the severity and the scope of each error type is readily apparent.

The technical score is based on errors in the model which will affect the operation of the model and its ability to correctly diagnose failures. These errors are critical and must be addressed as soon as they are found; before any further model development or review is performed.

The second part of the score, practices/conventions, refers to errors in the model which may affect the use of the suite of NASA developed tools (described in Sec. III) which extend and facilitate the use of TEAMS. These errors may also cause inaccuracies in the reports generated by the tools. Errors of this nature, are generally indicated by a failure of the model to follow the guidelines of the TMCP and may negatively impact the unit testing and validation review.

The last part of the score, cosmetic, refers to errors which will not affect the operation of the model or the tools, but which are inconsistent with the TMCP and may make the model more difficult for other modelers to understand. These errors are of the least importance, and may be due to a modeler's style preferences, but every effort should be made to ensure that the model is consistent and clear in order to aid in operation, integration and future development.

For most effective use, model implementation verification should be performed concurrently with model development. Specifically, models should be verified at multiple stages including low-level, internal reviews of FFMs 1) in generic component form, 2) in instantiated component form, 3) at the system or subsystem level, and 4) at high-level, external verification review of the full system or subsystem FFM.

1. Low-Level Verification of Individual Component Modules

Before a system level FFM is assembled from the component FFMs, the model implementation of each individual component should be independently verified. Early and thorough verification at this stage allows modeling errors to be more easily found and corrected and prevents the propagation of errors during system or subsystem level FFM integration and development.

FFM systems or subsystems are assembled primarily from instantiated generic FFM components or uniquely developed FFM components. Generic FFMs, such as the one shown in Fig. 1, have been developed to streamline the modeling process. Common parts and sensors, such as valves and temperature transducers, are developed in generic form and used to populate a model library. In generic form, the model elements are assigned a generic name, for instance a failure mode might be named Subsystem_Schematic-ID_Hyd-Fluid-Leak_FMEA-ID, following the conventions set forth in the TMCP. When the higher level system or subsystem is being modeled, the generic components are instantiated based on the specific subsystem, schematic and FMEA identification numbers and other identifying information of the component being modeled. In generic form, the components contain all potential failure and operational modes, both of which can be altered during the instantiation process to emulate the parameters of the actual component being modeled. This generic library allows common components to be developed and verified a single time instead of requiring independent development and verification of very similar FFMs.

The verification of the model implementation of generic, instantiated and uniquely developed FFM components should cover all of the categories listed in Table 1. A list of errors found in each category should be complied and error scores assigned based on the type and severity of each error. Suggested scores are given in the VV&A plan, but the specific scores for each FFM error should be determined by the model developers and/or project leads based on the type, complexity, size, and purpose of the FFM being developed. These scores are useful during the development process to correct the models and during the review process to determine the credibility of the models.

Table 1. Low-level verification categories for individual component modules.

Verification Categories	Technical	Practices/Conventions	Cosmetic
Model Level	Missing module Extraneous module	Incorrect name Incorrect hierarchy label	Incorrect appearance
Arcs (Links)	Missing/extraneous arc Incorrectly connected arc Missing/extraneous port Missing/extraneous failure effect	Missing port label	Incorrect appearance Incorrect port label
Failure Mode (FM)	Missing/extraneous FM Missing/incorrect failure effect Extraneous failure effect Incorrect tech label	Incorrect hierarchy label Incorrect FM name Inconsistent FMEA ID	Incorrect appearance
Testpoint (TP) /Test	Missing TP Incorrect TP location Missing Test Failure effect not detected by test Extraneous failure effect detection	Incorrect TP name Extraneous TP Incorrect test name Incorrect failure effect name Incorrect test label	Incorrect TP appearance
Mapper	Missing mapper Failure effect not mapped Incorrect failure effect transition Incorrect tech label	Incorrect name Failure effect assigned to mapper Incorrect mapper assigned to module Incorrect hierarchy label	Incorrect appearance

2. Low-Level Verification of Instantiated and System or Subsystem FFMs

Once the individual component models have been instantiated and integrated into the full system or subsystem model, it is necessary to verify the results of the instantiation process, the inter-module connections, any newly added model elements and the overall hierarchical structure.

If the instantiated components were previously verified in generic form, then the model implementation verification of the instantiation should cover only the categories shown in the Instantiation row of Table 2. The purpose of this verification is to confirm that the instantiation was performed correctly and that errors were not added to model elements or to hierarchy dependent names.

Table 2. Low-level verification categories for instantiated and system or subsystem level FFMs.

Verification Categories	Technical	Practices/Conventions	Cosmetic
Instantiation	Module names Sub-module names Failure mode names Failure effects Tests	Testpoint names	
System or Subsystem	Non-symmetric model Poor management of redundancy Inconsistent interfaces Incorrect level of model detail Inappropriate modeling assumptions	Unclear notes Missing or inappropriate references Poor model breakdown	Poor arc management

Multiple component models as well as modeling elements (e.g. testpoints, arcs, etc.) can appear at the system or subsystem level, as shown in Fig. 2, and the model may also contain many layers of hierarchy. At this stage of the verification process, the model implementation of any unverified component modules in the system or subsystem level FFM and of any newly added modeling elements should be verified according to the categories in Table 1. In addition, the overall structure, integration, and representation of the system or subsystem FFM should be verified according to the categories in the System or Subsystem row of Table 2. These categories are more ambiguous than those previously listed, but are important for a high quality, accurate and representative FFM. It is recommended that a highly experienced FFM developer perform this portion of the verification.

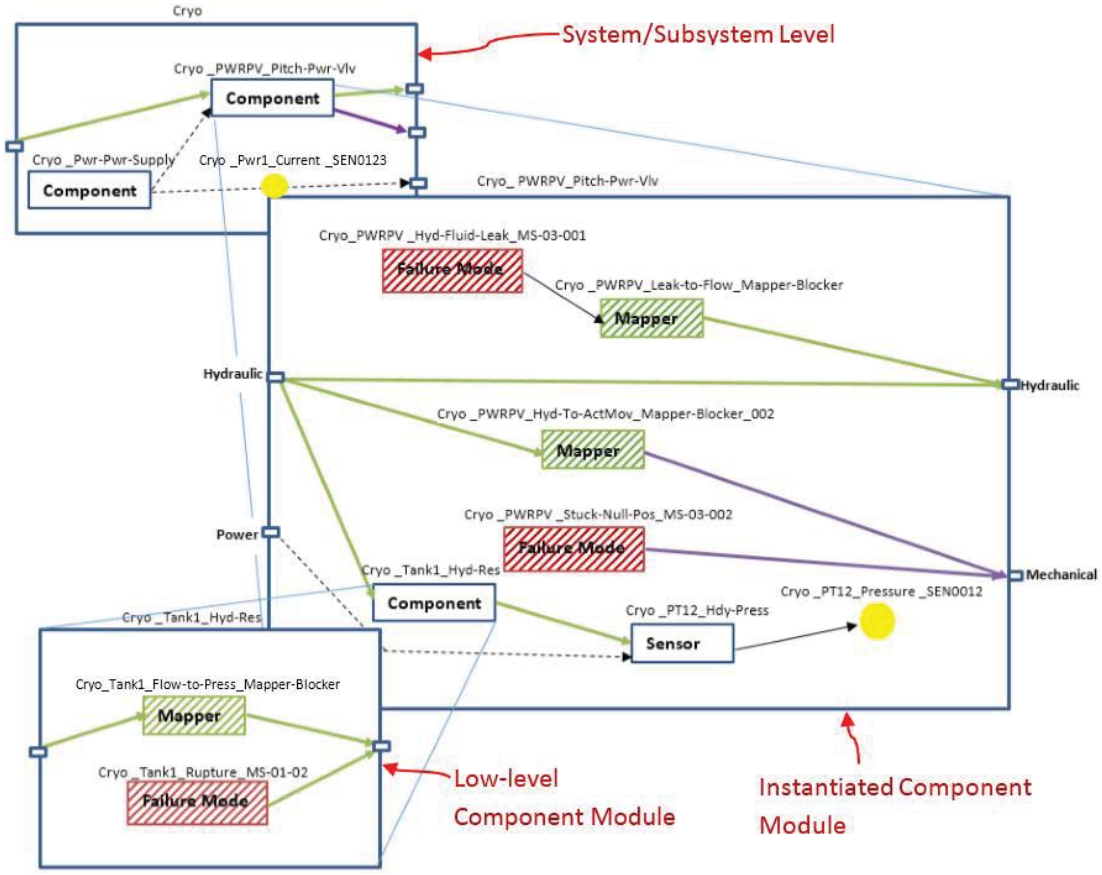


Figure 2. FFM with three hierarchy levels of instantiated modules.

3. High-Level Verification of Systems or Subsystems

Once the internal, low level reviews have been completed on the full system model, an external high-level review is necessary if the verification is to fulfill the accreditation requirements set forth in NASA-STD-7009. This verification should be performed with a Subject Matter Expert (SME) and should only be performed once all errors in the technical error category from the previous reviews have been corrected. The purpose of this model implementation verification review is to confirm that the configuration of the model and of the propagation paths of failures correctly reflect the physical system's structure and operation. To that end, TEAMS specific modeling practices and the model's adherence to proper conventions are not verified at this level.

The verification of the model should be performed jointly by the SME and an FFM developer and cover all of the categories in Table 3.

Table 3. High-level verification categories for systems or subsystems.

Verification Categories	
Modules	Missing module Extraneous module
Arcs (Links)	Missing/extraneous arc Incorrectly connected arc Missing/extraneous port Missing/extraneous failure effect
Failure Mode (FM)	Missing/extraneous FM Missing/incorrect failure effect Extraneous failure effect Incorrect tech label or system mode
Testpoint (TP) /Test	Missing TP Missing Test Failure effect not detected by test Extraneous failure effect detection
Mapper	Missing mapper Failure effect not mapped Incorrect failure effect transition
System/ Subsystem	Incorrect assumptions Inappropriate level of model detail

Any errors or poor assumptions that are found during the high level review should be corrected before the model is put into use. However the uncorrected score produced by the high level review should be used as the basis for the verification component of the accreditation score[2].

4. Model Implementation Verification Scoring

As previously mentioned, the specific penalties or scores given to each type of error should be determined by the model developers and/or project leads based on the nature of the FFM being developed. The scores assigned for errors in the initial, low-level reviews should be used primarily to assess the level of completeness and correctness of a model and to indicate areas of the model that require more development or debugging before the model is ready for the next stage of development or review.

The error scores assigned in the final, high-level review should be used to calculate the High Level Model Implementation Score shown in Eq. (1). This score can be used, in part, to access the credibility of the model according to NASA-STD-7009 and is meant to provide project stakeholders with an indication of the reliability of the model at the time that it was considered correct and complete according to the FFM developers' low-level, internal reviews.

$$High\ Level\ Model\ Implementation\ Score = \frac{Sum\ of\ Individual\ Error\ Scores}{Total\ Possible\ Error\ Score} \quad (1)$$

The denominator of Eq. (1) should be calculated by summing the total number of errors possible for every model element in the FFM that is listed in Table 3 along with an estimation of the errors possible for the system/subsystem verification category. The exact values assigned for the error scores do not matter as long as they are consistent across the model and scaled to equate to the severity of the error.

B. Dynamic Verification

Dynamic FFM verification is the actual exercising of the model in order to verify the diagnostic. Rigorous systematic testing must be performed to verify that each failure mode and test in the FFM is properly represented and connected. This testing process needs to be conducted for each system configuration represented in the model. Regression testing must also be performed if any changes are made to the model after the verification process has been completed.

While there is substantial overlap between model verification and validation, the dynamic verification techniques presented in this paper, especially unit testing, are frequently considered to be part of the validation process. However, NASA-STD-7009 specifies that unit and regression testing are "key aspects" of verification and therefore, for the purposes of this paper, unit and regression testing will be treated as aspects of verification.

1. Dynamic Verification via Unit Testing

In the current FFMs developed in TEAMS, failure modes represent single independent failures. Combined failures where one failure impacts the effects of another failure or precipitates another failure are not represented, unless they are modeled as a single failure mode. For example, given a hydraulic pump failure that causes a high fluid pressure output and a hydraulic fluid leakage that causes a pressure drop, this would be difficult to consider in the FFM as two independent failure modes are occurring. What would the combined effect be of high and low pressures from the two failure modes? But one could model a combination of failure modes as a single failure mode with a defined set of failure effects. Accepting this failure modeling constraint, testability analysis can be performed on the model where individual failure modes are invoked one at a time and the trace of tests detecting the effect generated.

Within TEAMS Designer a testability analysis cycle can be performed which systematically activates each available failure mode and records the available tests which detect the effects cascaded from that failure mode along the propagation paths. These results are output to a dependency matrix (see Fig. 3) where the columns are the tests and the rows are the failure modes. Where they intersect and a '1' is entered indicates the test detects an effect from that failure mode.

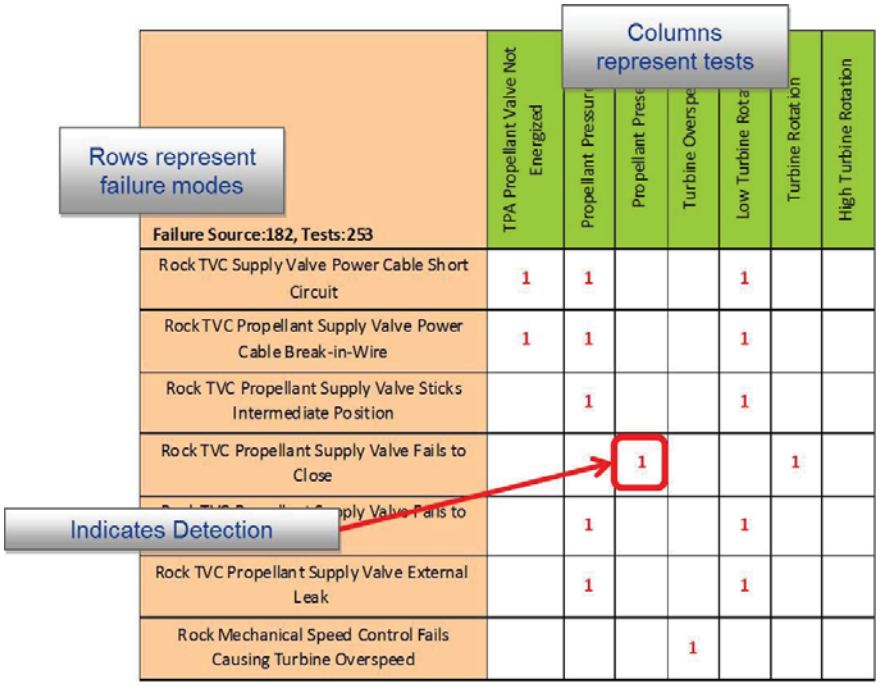


Figure 3. Example D-Matrix (Diagnostic Matrix) generated by the TEAMS Designer software.

The concept of building FFMs under recent NASA projects has been to build small unit or component models, frequently in generic form, and integrate the instantiated models into larger system or subsystem models. The step-wise model development offers the potential for model reuse and reduced future model development. As with the static verification, dynamic verification is required at the unit or component level and repeated at each model integration step.

FFM unit testing, verifying the smallest testable portion of the overall system model, confirms that there is an appropriate local relationship between every failure mode and every test in the component model. This testing verifies that failure effects produced by, or introduced to, individual components are detected by the expected internal component sensors and/or propagated out of the component through the correct pathways. Forward testing, confirming that each failure mode is detected by the appropriate test(s), and backward testing, confirming that each test detects the appropriate failure mode(s), are necessary to confirm expected detection coverage at the component level. The analysis should be conducted at each operational configuration to verify correct behavior of the model under each configuration.

FFM integration testing can be performed at various stages of the FFM model integration process. Anytime multiple component models or subsystem models are connected, these tests could be conducted. Forward and backward testing should be applied, as well as assessing the failure isolation capability of the larger modeled system.

Failure isolation assessment determines the diagnostic engines ability to distinguish between possible failure modes based on the available test suite. Integration testing should verify that, at subsystem and system levels, failure effects propagate correctly throughout the model via the connections between model components.

2. Dynamic Verification via Regression Testing

FFM regression testing is used to confirm that changes made after the model has undergone verification via model implementation, unit and integration testing do not have any unintended consequences. Small or minor changes to the FFM can be evaluated using a comparison between the diagnostic test results of the original and altered models. This comparison should confirm that newly alter model impacts the diagnostic assessment as intended. For wholesale changes in the model, the verification should be cycled back and performed at least at the integration level and possibly at the unit level depending on the extent of the model updates.

Complete verification of an altered FFM should include a full comparison of the D-matrices, Fig. 3, produced by the original and the altered models. All detections, indicated by a '1', or lack of detections between the unaltered portions of the model should be compared to confirm that the detections have not been unexpectedly altered.

III. Software Tools for More Efficient Verification of FFMs

Comprehensive verification of a complex system is a critical, yet highly resource intensive process. As such, significant research has been performed in order to reduce the time, effort and errors associated with the verification process while improving the efficiency, accuracy, and consistency of verification and streamlining the reporting of the verification results. Methods resulting from this research have included: focusing on only the most critical aspects of the model[8], selecting only specific verification, validation or accreditation activities to perform[9] and only considering specific classes of failures[10]. As the entirety of the FFM models utilized by NASA are required to be comprehensively verified for all types of errors using all required verification methods, three tools[11,12] have been developed to automate this process. These tools, developed at NASA's Glenn Research Center, include the VERification Analysis (VERA) Tool, which partially automates model implementation verification; the Extended Testability Analysis (ETA) Tool[13], which performs unit testing; and the D-Matrix Comparator, which performs regression testing between two versions of a model. The development and operation of the first two tools, VERA and ETA, will be described in this paper. For more information on the D-Matrix comparator, please see Ref. 12.

A. The VERification Analysis (VERA) Tool

For all but the most trivial models, an exhaustive model implementation review of the FFM is extremely time consuming and is as prone to human error as the development of the original model. Therefore, a program which automatically compiles a list of all the elements in an FFM and programmatically verifies (to the extent possible) that they follow the established conventions and practices has been developed. This program, VERA, is able to produce a comprehensive list of model elements and to verify approximately half of the low-level model implementation, based on the verification categories from the VV&A[5] plan listed in Table 1. Performing model implementation verification manually typically takes on the order of hours (for small models) to days or weeks (for larger models), while the VERA tool performs approximately half of the verification in a few seconds. The portion of the model implementation verification that the VERA tool performs is also the portion that humans are notoriously poor at, including checking for specific syntax errors in model element names. Seemingly minor syntactical errors, such as the substitution of a '0' for an 'O' in failure mode name, could cause the model to incorrectly diagnosis a failure and it is therefore critical that such errors be found and corrected. The VERA tool reliably detects these types of errors and provides a framework for a model reviewer to note system errors that the tool is unable to analyze, such as missing model elements or failure effects that are generated by an incorrect component.

1. The VERA Tool's Operation

VERA was written in Visual Basic for Applications and utilizes the Java based 'Batch Editor' tool developed at NASA's Ames Research Center. The Batch Editor [12] was developed to work with the underlying file structure of the FFM; extracting and modifying the model external from the TEAMS Designer interface. The VERA tool utilizes the extracted FFM information in order to inspect the model's structure and produces a comprehensive list of every element within the FFM, including all component modules, arcs/links, failure modes, failure effects, tests, and mappers. These elements are then analyzed by the tool to confirm that they:

- correctly follow NASA approved modeling conventions and practices[6]
- have the appropriate relationship with their parent and child elements

Specifically, the VERA tool generates an MS Excel workbook which contains all of the components of the verification categories from Table 1. A separate spreadsheet is created in the workbook for each verification category and every model element from each category is listed along with all pertinent information about that element that the VERA tool is able to collect from the extracted FFM information. In Table 4, verification components are listed under 'automated,' if they are analyzed and scored by the VERA tool, and listed under 'manual,' if they require human analysis and scoring.

Table 4. VERA scoring capabilities of code implementation verification categories.

Verification Category	Automated	Manual
Model	Incorrect appearance Incorrect name Incorrect hierarchy label	Missing module Extraneous module
Arcs (Links)	Missing port label Incorrect appearance	Missing/extraneous arc Incorrectly connected arc Missing/extraneous port Missing/extraneous failure effect Incorrect port label
Failure Modes (FM)	Incorrect hierarchy label Incorrect FM name Inconsistent FMEA ID Incorrect appearance	Missing/extraneous FM Missing/incorrect failure effect Extraneous failure effect Incorrect tech label
Testpoint (TP) /Test	Incorrect TP appearance Incorrect TP location Incorrect TP name Incorrect test name Incorrect test label Incorrect failure effect name	Missing TP Missing Test Failure effect not detected by test Extraneous failure effect detection Extraneous TP
Mappers	Incorrect tech label Incorrect name Failure effect assigned to mapper Incorrect mapper assigned to module Incorrect hierarchy label Incorrect appearance	Missing mapper Failure effect not mapped Incorrect function transition

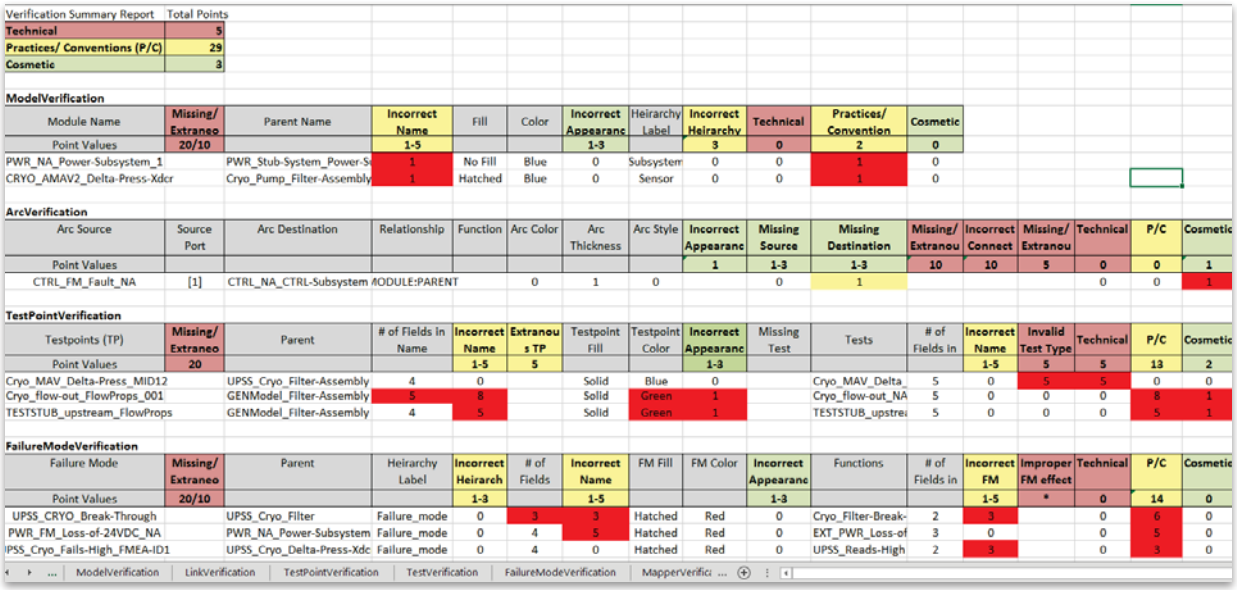
At this point, the VERA tool is not capable of analyzing or scoring the higher-level verification categories listed in Tables 2 and 3. However, while the VV&A plan specifies that FFMs should undergo code implementation verification at the component level, before they are integrated into system or subsystem level models, the VERA tool can still be used to analyze and verify integrated systems and subsystems and is not limited by the size or complexity of the model.

2. The VERA Tool Scoring and Verification Report

Upon completion of its operation, the VERA tool produces a report listing all of the elements in the model, all detected errors in the model, the location of each error, and it assigns a score for each error based on its severity. This report can be used to document the verification, as an outline with which the additional model implementation categories can be verified, and as a guide to correcting the detected errors.

The current error scoring penalties included in the VERA tool are based on the VV&A plan. However, they can be modified based on the requirements of the FFM under development.

A portion of a Verification Report produced by the VERA tool for a sample model is shown in Fig. 4. At the top of the report is the total verification score for the model, divided into technical practices/conventions, and cosmetic sections. The rest of the report lists each of the verification categories and all of the model elements that contain errors from that category.



Verification Summary Report	Total Points
Technical	5
Practices/ Conventions (P/C)	29
Cosmetic	3

ModelVerification

Module Name	Missing/ Extraneo	Parent Name	Incorrect Name	Fill	Color	Incorrect Appearanc	Heirarchy Label	Incorrect Heirarchy	Technical	Practices/ Convention	Cosmetic
Point Values	20/10		1-5			1-3		3	0	2	0
PWR_NA_Power-Subsystem_1		PWR_Stub-System_Power-S	1	No Fill	Blue	0	Subsystem	0	0	1	0
CRYO_AMAV2_Delta-Press-Xdcr		Cryo_Pump_Filter-Assembly	1	Hatched	Blue	0	Sensor	0	0	1	0

ArcVerification

Arc Source	Source Port	Arc Destination	Relationship	Function	Arc Color	Arc Thickness	Arc Style	Incorrect Appearanc	Missing Source	Missing Destination	Missing/ Extraneou	Incorrect Connect	Missing/ Extraneou	Technical	P/C	Cosmetic
Point Values								1	1-3	1-3	10	10	5	0	0	1
CTRL_FM_Fault_NA	[1]	CTRL_NA_CTRL-Subsystem	MODULE:PARENT		0	1	0		0	1				0	0	1

TestPointVerification

Testpoints (TP)	Missing/ Extraneo	Parent	# of Fields in Name	Incorrect Name	Extraneous TP	Testpoint Fill	Testpoint Color	Incorrect Appearanc	Missing Test	Tests	# of Fields in	Incorrect Name	Invalid Test Type	Technical	P/C	Cosmetic
Point Values	20			1-5	5			1-3				1-5	5	5	13	2
Cryo_MAV_Delta-Press_MID12		UPSS_Cryo_Filter-Assembly	4	0		Solid	Blue	0		Cryo_MAV_Delta_	5	0	5	5	0	0
Cryo_flow-out_FlowProps_001		GENModel_Filter-Assembly	5	8		Solid	Green	1		Cryo_flow-out_NA	5	0	0	0	8	1
TESTSTUB_upstream_FlowProps		GENModel_Filter-Assembly	4	5		Solid	Green	1		TESTSTUB_upstrea	5	0	0	0	5	1

FailureModeVerification

Failure Mode	Missing/ Extraneo	Parent	Heirarchy Label	Incorrect Heirarch	# of Fields	Incorrect Name	FM Fill	FM Color	Incorrect Appearanc	Functions	# of Fields in	Incorrect FM	Improper FM effect	Technical	P/C	Cosmetic
Point Values	20/10			1-3		1-5			1-3			1-5	*	0	14	0
UPSS_CRYO_Break-Through		UPSS_Cryo_Filter	Failure_mode	0	3	3	Hatched	Red	0	Cryo_Filter-Break-	2	3		0	6	0
PWR_FM_Loss-of-24VDC_NA		PWR_NA_Power-Subsystem	Failure_mode	0	4	5	Hatched	Red	0	EXT_PWR_Loss-of	3	0		0	5	0
PSS_Cryo_Fails-High_FMEA-ID1		UPSS_Cryo_Delta-Press-Xdc	Failure_mode	0	4	0	Hatched	Red	0	UPSS_Reads-High	2	3		0	3	0

ModelVerification | LinkVerification | TestPointVerification | TestVerification | FailureModeVerification | MapperVerifici ...

Figure 4. Portion of a Verification Report produced by the VERA tool.
This figure should be viewed in color as the error headings are colored to indicate their type.

Every model element that has one or more errors is listed, along with all pertinent information about that element that could be assembled by the VERA tool from the model files and the score(s) assigned to the error. For instance, under the model verification category, the modules are listed along with the names of their parents, their fill, color, and hierarchy labels – all under the grey headings. The errors are listed according to the error type; red headings indicate technical errors, yellow heading indicate practice/convention errors, and green headings indicate cosmetic errors. The scores for all errors that can be analyzed by the VERA tool are listed and the sums for each error type are totaled in the last three populated cells. The score for errors that cannot be analyzed by the VERA tool are left blank for the human reviewer to complete.

While the verification report in Fig. 4 only lists the model elements that contain errors, a complete list of all model elements in the entire FFM is created in separate spreadsheets - one for each verification category. If error scores are manually added to the model elements in the verification category spreadsheets, the model element and score will automatically be added to the Verification Report.

B. Extended Testability Analysis (ETA) Tool

Verification via unit testing can be supported using the ETA Tool, which is a software package developed at NASA's Glenn Research Center[11,13]. The ETA Tool extracts information from the TEAMS Designer analysis output and the associated diagnostic model to provide a detailed set of reports highlighting aspects of the system's diagnostic performance. Adherence of the FFMs to TMCP[6] facilitates the extraction of information from the model by the ETA Tool. The ETA Tool allows the user to select from several analysis reports, each report providing information about a specific aspect of the FFM diagnostic performance. The user should review the reports with the SME and resolve any discrepancies discovered. Key reports for the ETA Tool used for FFM verification are: detectability, test utilization and failure isolation.

The Detectability report supports an assessment of the detection coverage for the available suite of tests. For each modeled failure mode, the report indicates all the tests that detect failure effects propagating from that failure mode. Fig. 5 provides an illustration of the detection information being reported for an individual failure mode. Each failure mode (shown in Fig 5. as FM1-3) is analyzed and the effects assigned to that failure mode are propagated along model paths until they intersect with tests (shown in Fig 5. as TP1-3) designed to detect those effects or indirect effects mapped from the original effects. The collection of the tests that detect a particular failure mode is the detection signature for that failure mode and is reported for each failure mode. Also included in the report and just as important is the list of failure modes not detected by any of the available tests.

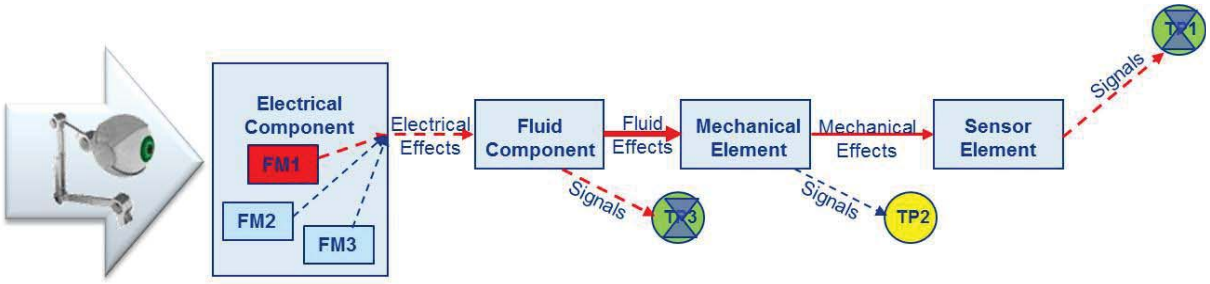


Figure 5. Illustration of the FFM assessment provided by the Detectability report.

In Fig. 5, failure mode FM1, highlighted in red, is propagating electrical failure effects along the path highlighted in red. These electrical effects are transformed to fluid and mechanical effects by the modules along the path. The resulting effects are detected by testpoints TP1 and TP3. Therefore, the Detectability report lists that FM1 is detected by TP1 and TP3.

The Test Utilization report provides the reverse perspective of the model from the Detectability Report, by supplying the failure modes detected by each test. Figure 6 illustrates the analysis perspective of this report. Any of the failure mode along the reverse failure effect propagation path that is detected by a particular test is reported for that test. This information supports the design justification of the sensor usefulness for the system from the diagnostic perspective. As with the Detectability report, the Test Utilization report includes a list of tests that provide no detection coverage.

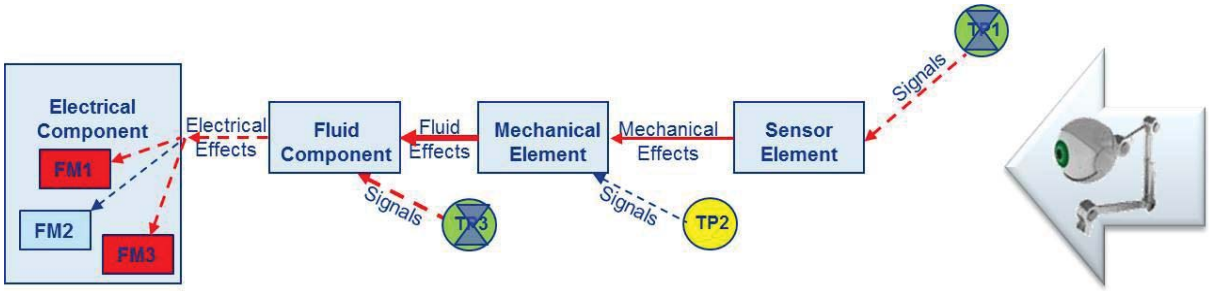


Figure 6. Illustration of the FFM assessment provided by the Test Utilization report.

In Fig. 6, testpoints TP1 and TP3 detect the failure effects produced by failure modes FM1 and FM3. Therefore, the Test Utilization report notes that TP1 detects FM1 and FM3 and TP3 also detects FM1 and FM3.

Another report available through ETA Tool is the Failure Mode Isolation report. This report provides groups of individual failure modes which have the same detection signature. Confirmation of the anticipated ambiguity among the failure mode modeled in the FFM, further verify the model's performance. Also, the assessment of the FFM ability to provide the required failure mode isolation capability can be made and system designers can begin to explore other potential sensors and tests that could resolve any indicated ambiguities discovered. Figure 7 illustrates two failure modes from different components that have the same detection signature and therefore are ambiguous with each other.

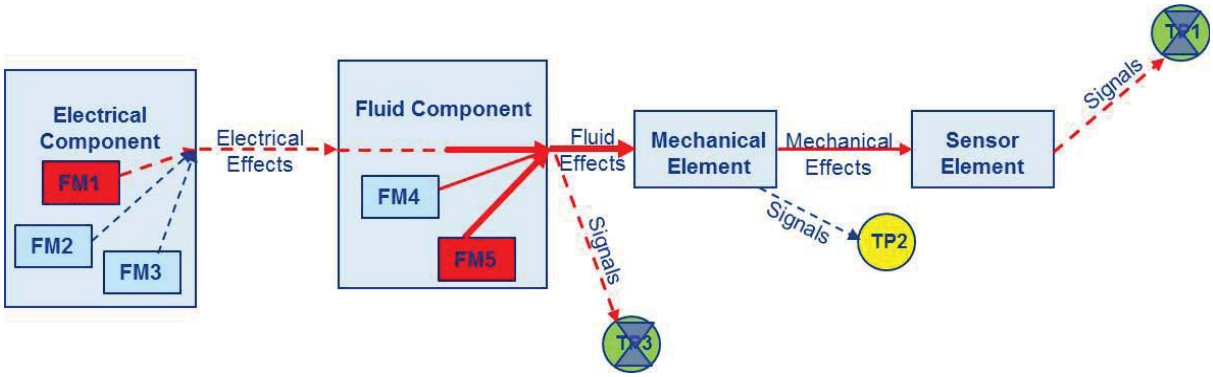


Figure 7. Illustration of a failure isolation assessment indicating two failure modes with same detection signature.

In Fig. 7, The 'Electrical Component' and 'Fluid Component' container modules represent physical elements in the system and contain their respective failure modes. Here failure modes FM1 and FM5, when active independently, are detected by the same tests in testpoints TP1 and TP3 and therefore are ambiguous with each other. Since they originate in separate components, those components are also ambiguous with each other.

If the FFM is modeled with local system modes and propagation path switches, then diagnostic assessments can be confirmed based on system configuration. This capability should be employed regardless of whether the eventual real-time FFM application takes advantage of the modeled system modes and switches or addresses model configuration in the model wrapper code. In addition, internal labels can be used in the model to 'turn-off' or disable failure modes and tests that would not be present under certain operating conditions or system configurations. These modeling practices will also provide cleaner analysis reports. Separate analyses would then be conducted at each system mode of interest with modeling practices that restrict applicable failure modes and tests for that configuration. Ultimately these results can be compared to the complete real-time FFM diagnostic implementation testing results to isolate diagnostic errors encountered.

For integrated testing of the FFM, another useful report is the Component Isolation Assessment report. This report will provide an indication of which failure modes within components are detectable and isolatable relative to the components of interest and is useful when the system has the requirement to provide a level of isolation to a particular set of components. The example displayed in Fig. 7 is not only ambiguous at the failure mode level, but also at the component level. The system as designed cannot distinguish between the failed electrical and fluid components. This level of diagnostic assessment can support verification launch operation and maintenance processes and requirements. Component isolation is not straight forward and often some failure modes in the component are detected and isolated, while others are not. Figure 8 shows a sample table from this report that displays in color failure modes that are undetected in red; detected but ambiguous with other considered components in orange; detected but ambiguous with other components not part of the analysis in yellow; and detected and isolated in green. With this level of detail, system engineers can make more informed decisions on operations and logistics, understanding the risks and limitations of those decisions.

Component Isolation Assessment for Components labeled as "LRU"

Legend Color Key

	Highlights Failure Modes that are Isolated, meaning they come from a group where this is the only component
	Highlights Failure Modes that are Detected, but not isolated with respect to components that don't include specified component candidates
	Highlights Failure Modes that are Detected, but not isolated with respect to the other specified component candidates
	Highlights Failure Modes that are Not Detected

Component	Number of Failure Modes	Number of Failure Mode Groups	Detected Failure Modes							
			Failure Mode Group	Failed Component			Failure Mode	FMEA Identifier	Criticality	Probability (MTTF [hrs])
				Location	Name	Schematic Identifier				
Pitch Turb Pump Assy	5	4	48	Pitch Turb Pump Assy	Pump	PMPP	Fails To Function	MS-SS-Hpump-02-001	1R	1e+006
			50	Pitch Turb Pump Assy	Pump	PMPP	Hyd Fluid Leak External	MS-SS-Hpump-02-002	1R	1e+006
			58	Pitch Turb Pump Assy	Shaft Speed	SS1	SF Bad Signal	MS-SS-HPump-08-001	1S	1e+006
			59	Pitch Turb Pump Assy	Turbine	TRBP	Fails To Function	MS-SS-HPump-01-001	1R	10000
				Pitch Turb Pump Assy	Turbine	TRBP	Prop Leak External	MS-SS-HPump-01-002	3	1e+006
Yaw Turb Pump Assy	5	4	40	Yaw Turb Pump Assy	Pump	PMPY	Fails To Function	MS-SS-Hpump-02-001	1R	1e+006
			42	Yaw Turb Pump Assy	Pump	PMPY	Hyd Fluid Leak External	MS-SS-Hpump-02-002	1R	1e+006
			53	Yaw Turb Pump Assy	Shaft Speed	SS2	SF Bad Signal	MS-SS-HPump-08-001	1S	1e+006
			54	Yaw Turb Pump Assy	Turbine	TRBY	Fails To Function	MS-SS-HPump-01-001	1R	1e+006
				Yaw Turb Pump Assy	Turbine	TRBY	Prop Leak External	MS-SS-HPump-01-002	3	1e+006

Figure 8. Sample portion of the Component Isolation Assessment report.

IV. Concluding Remarks

Verification of models and simulations is a complex process with the conflicting priorities of increasing the completeness and consistency of verification while decreasing the time and resources necessary to complete the verification process. To fulfill the first priority, this paper presents the three methods used to comprehensively verify an FFM (Functional Fault Model), specifically model implementation verification, unit testing and regression testing. These methods include both static and dynamic verification processes, thus ensuring that the model is fully verified. To address the second priority, this paper presents the automated tools, Verification Analysis and Extended Testability Analysis, whose use increased the efficiency and accuracy of verification while greatly decreasing the time and resources necessary to complete this process. The complete verification process presented in this paper can be used to determine the verification evidence credibility level required for accreditation of FFMs by NASA-STD-7009, NASA Standards for Model and Simulations. However, this process is also useful as a stand-alone verification method for models that are being developed according to other standards.

The methods and tools developed in this paper apply to only a very specific subset of models and simulations: FFMs developed in the Testability Engineering And Maintenance System (TEAMS) Designer that follow the TEAMS Modeling Conventions and Practices. While these methods and tools are not generally applicable to all simulation and model verification, it is hoped that the material developed and presented in this paper will provide ideas and inspiration to other modelers attempting to develop verification methods and tools for other applications.

References

[1]TEAMS®-Designer, Testability Engineering And Maintenance System, Software Package, Ver. 12.1. QualTech Systems Incorporated, East Hartford, CT. http://www.teamsqsi.com [cited 10 November 2014].

[2]NASA-STD-7009, NASA Standard for Models and Simulations; July, 2008; https://standards.nasa.gov/documents/detail/3315599 [cited 17 July 2014].

[3]Sargent, R. G., "Verification, validation, and accreditation: verification, validation, and accreditation of simulation models," *Proceedings of the 32nd conference on Winter simulation*. Society for Computer Simulation International, Orlando, FL, 2000, pp.50-59.

[4]Sargent, R. G., "Verification and validation of simulation models," *Proceedings of the 37th conference on Winter simulation*. Society for Computer Simulation International, Orlando, FL, 2005, pp. 130-143.

[5]K0000190527-PLN, Functional Fault Modeling and Fault Isolation Verification, Validation and Accreditation Plan, National Aeronautics and Space Administration, 2014.

[6]K0000190582-GEN, Testability Engineering and Maintenance System (TEAMS) Modeling Conventions and Practices, National Aeronautics and Space Administration, 2014.

[7]Ferrell, B., Lewis, M., Perotti, J., Oostdyk, R., and Brown, B., "Functional Fault Modeling Conventions and Practices for Real-Time Fault Isolation," *SpaceOps 2010 Conference Delivering on the Dream*, Huntsville, AL, 2010.

[8]Liu, F., and Yang, M., "An optimal design method for simulation verification, validation and accreditation schemes," *Simulation*, Vol. 85, No. 6, 2009, pp. 375-386.

[9] Muessig, P. R., Laack, D. R., and Wrobleski, J. W. "Optimizing the selection of VV&A activities: a risk/benefit approach," *Proceedings of the 1997 Winter Simulation Conference*, Atlanta, GA, 1997, pp. 60–66.

[10]Lutz, R. R., "Targeting safety-related errors during software requirements analysis," *SIGSOFT Software Engineering Notes*, Vol. 18, No. 5, New York, NY, 1993, pp. 99-106.

[11]Maul, W., Melcher, K., Chicatelli, A. and Johnson, S., "Application of Diagnostic Analysis Tools to the Ares I Thrust Vector Control System," *AIAA InfoTech@Aerospace Conference*, Atlanta, Ga, 2010. [NASA TM-2010-216333, AIAA 2010-3451]

[12]Barszcz, E., Robinson, P., and Fulton, C., "Tools Supporting Development and Integration of TEAMS Diagnostic Models," *AIAA InfoTech@Aerospace Conference*, St. Louis, MO, 2011. [AIAA 2011-1639]

[13] Maul, W., Fulton, C. and Melcher, K., "Extended Testability Analysis Tool User Guide," *AIAA InfoTech@Aerospace Conference*, St. Louis, MO, 2011. [AAIA 2011-1637]