Towards Verification of Operational Procedures using Auto-Generated Diagnostic Trees

Tolga Kurtoglu[1], Robyn Lutz[2], and Ann Patterson-Hine[3]

[1] *Mission Critical Technologies @ NASA Ames Research Center, Moffett Field, CA, 94035, USA*
tolga.kurtoglu@nasa.gov

[2] *Jet Propulsion Laboratory/Caltech, Pasadena, CA, 91109, and Iowa State University, USA*
robyn.r.lutz@jpl.nasa.gov

[3] *NASA Ames Research Center, Moffett Field, CA, 94035, USA*
ann.patterson-hine@nasa.gov

ABSTRACT

The design, development, and operation of complex space, lunar and planetary exploration systems require the development of general procedures that describe a detailed set of instructions capturing how mission tasks are performed. For both crewed and uncrewed NASA systems, mission safety and the accomplishment of the scientific mission objectives are highly dependent on the correctness of procedures. In this paper, we describe how to use auto-generated diagnostic trees from existing diagnostic models to improve the verification of standard operating procedures. Specifically, we introduce a systematic method, namely the Diagnostic Tree for Verification (DTV), developed with the goal of leveraging the information contained within auto-generated diagnostic trees in order to check the correctness of procedures, to streamline the procedures in terms of reducing the number of steps or use of resources in them, and to propose alternative procedural steps adaptive to changing operational conditions. The application of the DTV method to a spacecraft electrical power system shows the feasibility of the approach and its range of capabilities.[*]

[*] This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

1 INTRODUCTION

The design, development, and operation of complex space, lunar and planetary exploration systems require the development of general procedures that describe a detailed set of instructions capturing how mission tasks are performed (Frank, 2008). These procedures include a complicated mix of software checks and calibrations, conditional commands, manual inputs and checks of console data, and inspection of physical equipment. For both crewed and uncrewed NASA systems, mission safety and the accomplishment of the scientific mission objectives are highly dependent on the correctness of the procedures. It is therefore imperative that these procedures are verified and validated before being used. However, the development, inspection and verification of these procedures remain a key technical challenge for various reasons.

First, the development of the standard operating procedures by system designers is currently labor-intensive and critically dependent on human expertise (Kortenkamp et al., 2008). The process generates thousands of pages of documentation and many opportunities for human error.

Second, operators using these procedures face the challenge of how to handle changes to system reconfiguration or system health conditions that can arise during real-time operations. Often, general procedures need to be adapted to specific mission conditions in a manner that is safe and effective. Moreover, procedures - especially those covering contingencies - are static in nature and only specify a single path to recovery. This non-redundant approach makes it difficult for operators and flight controllers to interpret appropriate next steps if a particular step in a procedure becomes invalid or non-applicable, e.g., because the resource to be used is unavailable.

Third, procedures concerning off-nominal scenarios only cover single point failures. Accordingly, each step in a procedure defines local checks and calibrations of physical parameters and data associated with a particular anomaly. This makes it difficult to respond to multiple faults in a system, or to single faults where the proper response demands integration of data from a large number of sensors and reasoning based on a multitude of data sources.

This paper introduces a systematic method, namely the Diagnostic Tree for Verification (DTV), developed with the goal of addressing the aforementioned three challenges. The basis of the DTV approach is to leverage knowledge and automated analyses readily available in model-based diagnostic systems. Specifically, the DTV method describes how to use auto-generated diagnostic trees from existing diagnostic system models to (1) check the correctness of the procedures, (2) streamline the procedures in terms of reducing the number of steps or use of resources in them, and (3) propose alternative procedural steps adaptive to changing operational conditions including those anomalies arising from multiple faults in a system. The application of the presented method to portions of a representative electrical power system (EPS), called the Advanced Diagnostic and Prognostic Testbed (ADAPT), demonstrates these capabilities.

The organization of the rest of the paper is as follows. Section 2 presents a review of related work in verification and validation of operational procedures. Section 3 describes the Diagnostic Tree for Verification method, the associated process and the evaluation metrics that can be used to measure the success of the proposed method. Section 4 introduces the Advanced Diagnostic and Prognostic Testbed that is used as a case example in this study. Section 5 discusses the application of the DTV method to the verification of a specific set of procedures developed for the ADAPT system. Section 6 summarizes the significance of the proposed approach and major challenges. Finally, Section 7 presents concluding remarks and an outlook for future work.

2 RELATED WORK

There are various techniques developed that can help verify and validate procedures.

Most current verification techniques are largely manual (inspection and reviews) and focus primarily on the conformance of command programs – scripts written for execution of procedures - to procedure definitions. Some advanced procedure authoring tools such as PRIDE (Kortenkamp et al., 2008) support syntax checking and can enforce syntax constraints.

Automated approaches to verification provide much needed support to mission operations. These approaches enable the verification of syntactic and semantic differences in procedure scripts, and simulation capabilities to validate the equivalence and correctness relations between different system representations (Brat et al., 2008).

Among these, static checkers verify that procedures are syntactically and semantically well written and structured. They are used to check for variable declarations, run-time errors, null pointers, operational bounds, order of procedure calls, etc. An example of a static analysis tool is the commercially available Polyspace C-verifier (Polyspace, 2008). Static checks, however, cannot catch all possible problems with procedures.

A more improved verification method, model checking, allows for the systematic exploration of the state space of a system that captures all possible behaviors of a system. As a result, model checkers can find errors in models like deadlocks, race conditions, or can verify whether a system reach into a desired state. Examples of model checkers used in space applications include LTSA (2009), and Java Path Finder (Visser et al., 2003).

Model checking has also been used outside of the engineering domain. For example, Damas et al. (2009) use model checking to analyze cancer treatment processes. The processes are described as guarded high-level message sequence charts that are then compiled into guarded Labeled Transition Systems. The focus of this research is on assembling clinical process fragments into a model for automated guard analysis and property verification rather than, as here, on using existing models to check and streamline operational procedures.

In this work, we are interested in moving beyond traditional verification of correctness and consistency of the procedures toward improved correctness. By improved correctness, we mean alternative, better ways to achieve tasks intended by standard operating procedures. To accomplish this, the DTV method seeks to exploit knowledge and automated analysis techniques applied for the diagnostic process by model-based diagnosis systems.

Model-based reasoning methods utilize a wide variety of engineering models as the foundation for representing diagnostic knowledge and developing algorithms that use this knowledge for fault detection and isolation. In parallel developments, different communities have found value in analytic state-based models, input-output transfer function models, fault propagation models, and quantitative physics-based models to develop online automated diagnostic software for monitoring and diagnosis of dynamical systems (Patterson-Hine et al., 2005).

Battery A Volts Low	Voltage sensor before battery CB shows low voltage	✓ **If downstream battery voltage measurement reads ok...** Battery A Out volts ≥ 21? • Battery A Out volts (element = E140)		
		Yes	**No**	
		1. **Suppress msg:** Battery A Volts Low 2. **Exit**	✓ **If the battery is connected to a load bus...** DistAA switch ON OR DistAB switch ON? • DistAA switch (element = ESH141A) • DistAB switch (element = ESH144A)	
			Yes	**No**
			Battery has insufficient voltage or is bad: put a different battery on the bus 1. **Perform:** "Change Power Source" (start at step 1)	**After solar arrays are deployed, can charge the battery** **Once on-orbit, go to:** "Charge Battery A" (start at step 1)

Figure 1. An excerpt from a procedure for a "battery voltage anomaly" in the ADAPT system.

3 DIAGNOSTIC TREE FOR VERIFICATION (DTV) METHOD

The DTV method seeks to exploit knowledge and automated analysis techniques applied for the diagnostic process by model-based diagnosis systems. These tools utilize information from the design phase, such as safety and mission assurance analysis, failure modes and effects analysis (FMEA), fault propagation models and testability analysis, and employ topological and analytical models of the nominal and faulty operations of a system for fault diagnosis, isolation, and recovery (Patterson-Hine et al., 2005). This information provides an independent perspective that the DTV method uses in order not only to verify the correctness and consistency of standard operating procedures, but also to improve the procedures themselves. The details of the proposed method are explained in the next sections, but first a brief overview of procedures is given.

3.1 Procedures

Franks (2008) defines procedures as: "A procedure is a detailed set of instructions specifying how a piece of equipment is operated, or a task is to be performed. Each step of a procedure may have conditions that must be satisfied before it can take place, and effects that must be understood when considering the implications on other steps of procedures. Procedure execution involves issuing commands to spacecraft, robots or systems; interpreting the responses of those systems; and choosing the next step in the procedure based on those responses. Procedures embody the engineering knowledge of the system or equipment involved in the tasks, and cover both nominal and off-nominal cases that arise." The procedures can be represented in human understandable format such as in PRL (Kortenkamp *et al.*, 2008), or machine-readable format such as in PLEXIL (Verma *et al.*, 2005).

In this research, we are focusing on two generic classes of procedures: procedures to diagnose/isolate a particular failure in a system, and procedures to recover from a failure in a system.

An example procedure is shown in Figure 1. This procedure describes how to isolate "a battery voltage anomaly" in the ADAPT system and how to recover from it by reconfiguring the system. The operational steps in the procedure include checks/tests/verification of the physical and software parameters of the system, commands to and from the system, and manual and automated actions for recovery.

3.2 DTV Modeling Environment: TEAMS

This section explains the modeling environment used by the DTV methodology.

The Testability Engineering and Maintenance System (TEAMS) tool suite (QSI, 2009) is the primary platform used for modeling by the DTV method. TEAMS is built upon the multi-signal modeling formalism (Deb *et al.*, 1995), which is a hierarchical modeling methodology where the propagation paths of the effects of a failure are captured using directed graphs. The model is based on structural connectivity or a conceptual block diagram of a physical system connected by links or paths. Software modules interfacing with the system are treated like any other hardware component, and can be included in the model. *Functions* describe attributes of system variables to be traced. The TEAMS modeling elements called test points are then added to the model. *Test points* represent the physical or computational locations of checks using sensors or sensor data as well as other means for observing a system. *Tests* are checks that look at the data from the sensors and make decisions

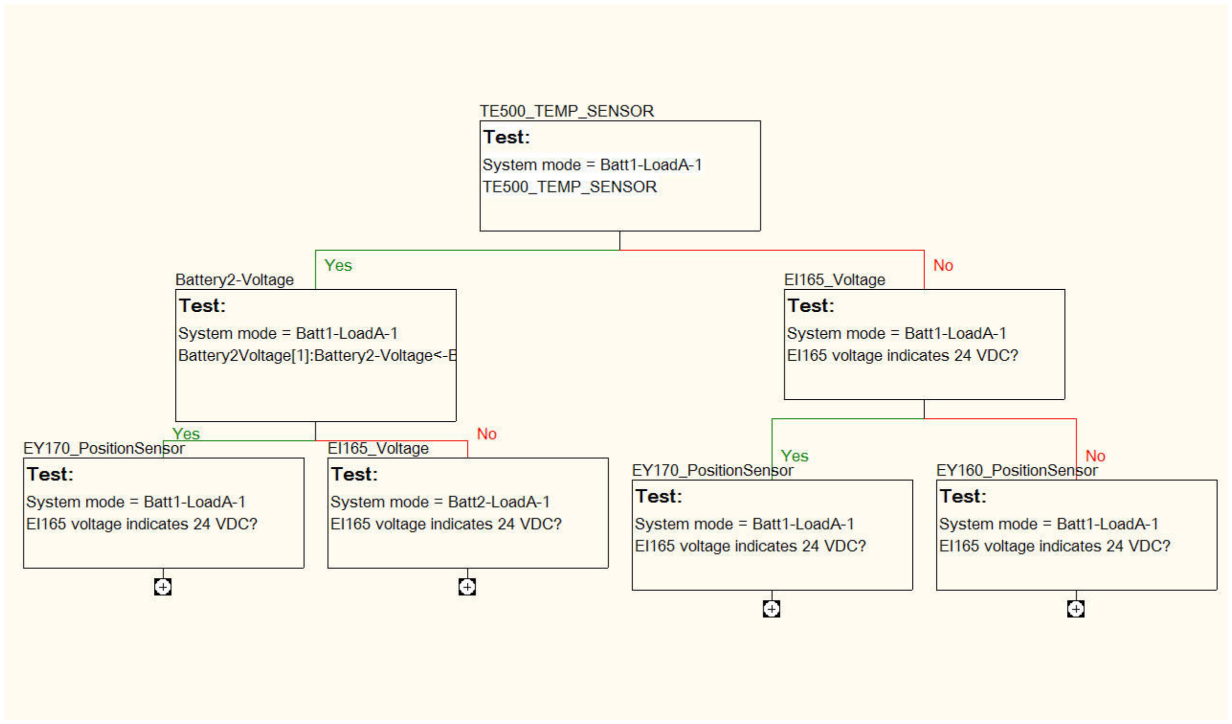

TE500_TEMP_SENSOR
Test:
System mode = Batt1-LoadA-1
TE500_TEMP_SENSOR
Yes
No
Battery2-Voltage
Test:
System mode = Batt1-LoadA-1
Battery2Voltage[1]:Battery2-Voltage<-E
EI165_Voltage
Test:
System mode = Batt1-LoadA-1
EI165 voltage indicates 24 VDC?
EY170_PositionSensor
Test:
System mode = Batt1-LoadA-1
EI165 voltage indicates 24 VDC?
EI165_Voltage
Test:
System mode = Batt2-LoadA-1
EI165 voltage indicates 24 VDC?
EY170_PositionSensor
Test:
System mode = Batt1-LoadA-1
EI165 voltage indicates 24 VDC?
EY160_PositionSensor
Test:
System mode = Batt1-LoadA-1
EI165 voltage indicates 24 VDC?

Figure 2. An excerpt from diagnostic tree for a “temperature anomaly” in the ADAPT system.

about system attributes associated with those measurements. This graph topology is then converted into a matrix representation describing the relationship between faults and test points for a given mode of the system. This representation contains the basic information needed to interpret test results and diagnose failures during operations. In addition, *actions* corresponding to recovery and maintenance tasks are included in the TEAMS model.

3.3 Diagnostic Trees

The DTV method uses existing TEAMS models to systematically explore the diagnostic trees that can be produced from the relationships between faults, tests, and actions in the model. *A diagnostic tree* describes a sequence of checks based on the results of prior checks. Different operational modes or configurations have different diagnostic trees since some faults are only possible, and some checks are only appropriate or available, in certain configurations or modes of operation.

For faults that cannot be detected and/or isolated, the diagnostic tree shows the set of indistinguishable failures, called *ambiguity groups*. Thus, all the nodes of the diagnostic tree at the top-level and the intermediate levels are tests in the model. The leaf nodes of the diagnostic tree are either fault-handling actions (e.g., remove/replace a failed part/component) that have been specified for each component in the model, faults that have been isolated but for which there is currently no recovery possible, or an ambiguity group of faults that cannot currently be detected and isolated by the available tests.

An example diagnostic tree is shown in Figure 2. This diagnostic tree describes how to isolate “a temperature anomaly” in the ADAPT system. Similar to the procedures, the steps in the diagnostic tree include checks/tests/verification of the physical and software parameters of the system, and manual and automated actions for recovery and maintenance.

3.4 Analysis of Procedures using Auto-Generated Diagnostic Trees

The diagnostic trees can be auto-generated from existing TEAMS models, thus making the DTV method easy to adopt. By dynamically exploring the model using the set of diagnostic trees generated, one can gain insight into the efficacy and efficiency of alternative fault-isolation and recovery paths.

By comparing the sequence of steps described in the procedure to isolate the causes of the anomaly and recover from it with the diagnostic paths of the diagnostic tree, we seek to find relationships between the two representations as illustrated in Figure 3.

Our current effort is aimed at finding any shorter sequence of tests and actions (i.e., a shorter path) that produces the same results (i.e., isolates the same fault, or recovers from a malfunction), for comparable or lesser cost in time or resources. This extends our previous work on three NASA applications (an unpiloted aerial vehicle, MER critical pointing software, and ADAPT) that showed how early

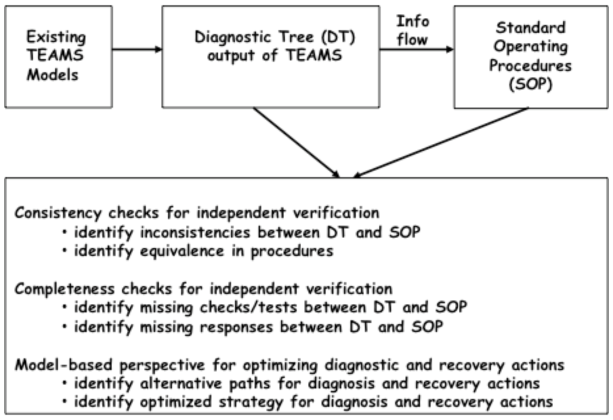

Existing TEAMS Models
Diagnostic Tree (DT) output of TEAMS
Info flow
Standard Operating Procedures (SOP)
Consistency checks for independent verification
• identify inconsistencies between DT and SOP
• identify equivalence in procedures
Completeness checks for independent verification
• identify missing checks/tests between DT and SOP
• identify missing responses between DT and SOP
Model-based perspective for optimizing diagnostic and recovery actions
• identify alternative paths for diagnosis and recovery actions
• identify optimized strategy for diagnosis and recovery actions

Figure 3. Overview of the developed DTV method.

consideration of potential anomalies using the diagnostic tree could help build in robustness for handling software contingencies (Lutz *et al.*, 2008; Lutz *et al.*, 2007; Lutz and Patterson-Hine, 2008a; Lutz and Patterson-Hine, 2008b).

This comparison is currently performed manually, however, we are working towards developing a representation that would capture the information contained within the procedures and diagnostic trees using a common language, such that the comparative analysis can be done in an automated fashion.

3.5 Metrics for Evaluation of the DTV Method

The details of the proposed method are explained in the next sections, but first a brief overview of procedures is given.
There are three types of metrics that one can use to evaluate the usefulness of the DTV approach:

1. Correctness

a. Path coverage: Determine whether the operational procedure covers all the paths in the diagnostic tree auto-generated by the model.

b. Branch coverage: Determine whether the operational procedure covers all the branches (i.e., includes all the tests) in the diagnostic tree auto-generated by the model.

2. Reduced complexity

a. Shorter path. Does the diagnostic tree identify an operational procedure that is equivalent in terms of isolating the same fault(s) as the operational procedure, but that contains fewer steps?

b. Fewer branches. Does the diagnostic tree identify an operational procedure that is equivalent in terms of isolating the same fault(s) as the operational procedure, but that contains fewer tests?

3. Improved efficiency

a. Resource usage. Does the diagnostic tree identify an operational procedure that is equivalent in terms of isolating the same fault(s) as the operational procedure, but that uses fewer resources. Of most interest are consumable resources e.g., manual steps (requiring human labor), power-cycling steps of a spacecraft instrument (where a maximum number of lifetime cycles will be allowed).

b. Reduced cost. This class of metrics is closely related to the resource usage. Since specific costs (financial, power, or duration) can be associated with specific tests. Construction of the diagnostic tree can be directed to use these costs and may be able to identify lower-cost alternatives.

c. Increased autonomy. If a high cost is assigned to tests requiring human-in-the-loop (e.g., where a switch must be thrown manually) and low cost to automated alternatives (e.g., software commanding a change in the switch position), the auto-generated diagnostic tree will seek to minimize the cost. Our hope (not realized in our current application) is that this may result in the tree displaying an alternative trouble-shooting strategy that offers increased opportunity for autonomy over an existing manual procedure.

d. Improved sensor and test placement. Because the model can be easily modified to add, delete, change, or move sensors, test points, and tests, the effects of alternative design decisions can be investigated. By comparing the resulting diagnostic trees with the original diagnostic tree, improvements in the operational procedures that would be feasible, e.g., with an additional sensor or test, may be identified.

4 CASE EXAMPLE: ADAPT – ELECTRICAL POWER SYSTEM TESTBED

In this section, we provide a brief overview of the ADAPT system and describe its major elements.

The Advanced Diagnostics and Prognostics Testbed (ADAPT) at the NASA Ames Research Center is a unique facility designed to test, measure, evaluate, and mature diagnostic and prognostic health management technologies. Reflecting the importance of electrical power systems (EPS) in aerospace (Button and Chicatelli, 2005; Poll et al., 2007), ADAPT provides a representative aerospace vehicle EPS that enables automated diagnosis in a complex domain. A simplified version of main functions and layout of the ADAPT power system are shown in Figure 4. The EPS can deliver power to various loads, which in an aerospace vehicle would include subsystems such as the avionics, propulsion, life support, and thermal management systems.

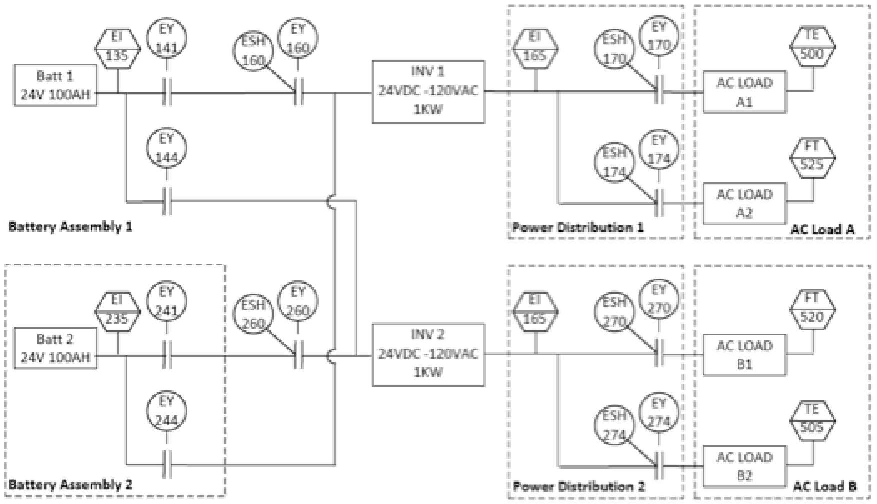


Figure 4. The simplified schematic of the ADAPT EPS System (Ghosal and Azam, 2008)

ADAPT contains elements common to many aerospace applications: power storage and power distribution. In the simplified version used in this study, the power storage consists of two battery modules. Either of the two batteries can be used to power either of the two load banks in the power distribution element. This design gives the ADAPT EPS basic redundancy and reconfiguration capability. Electromechanical relays are used to route the power from the sources to the batteries, and from the batteries to the loads. An inverter converts the DC battery input to AC output. Circuit breakers are located at various points in the distribution network to prevent overcurrents from causing unintended damage to the system components.

A data acquisition and control system sends commands to and receives data from the EPS. Testbed operator stations are integrated into a software architecture that allows for nominal and faulty operations of the EPS, and includes a system for logging all relevant data. The instrumentation allows for monitoring of voltages, currents, temperatures, switch positions, light intensities, and AC frequencies, and includes over 100 sensors. (More information on the ADAPT testbed can be found in (Poll et al., 2007)). Later in this paper, we use models and examples from the ADAPT system to illustrate the use and feasibility of the DTV methodology.

5 PRELIMINARY RESULTS

To demonstrate the feasibility of the DTV technique, we are applying it to portions of a representative electrical power system (EPS), previously introduced in Section 4.

The TEAMS model developed for this purpose consists of basic hardware components including batteries, relays, circuit breakers, a set of operational loads, etc., and sensors measuring the physical characteristics of the system. In addition, the model includes the software architecture of the system, mainly a data acquisition and control system that sends commands to and receives data from the testbed.

The operational procedures for the EPS system are modified from an advanced caution and warning system developed as an interface concept for a crewed vehicle (McCann et al., 2006).

In running these analyses, we first identify the symptoms (or observable states) of the system that correspond to off-nominal conditions in the system. An example is a “relay failure”. We then set particular symptoms to be active in the TEAMS model by using the user interface menu provided for all symptoms defined in the model. This automatically generates a diagnostic tree by forcing the analyzed symptom to be the root node of the tree. As described before, the diagnostic trees capture a sequence of tests, and checks that needs to be performed in order to diagnose/isolate the fault that is causing the analyzed symptom. The isolated fault(s) may also be annotated with recovery actions in the model, which then will also appear in the appropriate intermediate or leaf nodes of the diagnostic tree.

Similarly, the procedures in (McCann et al, 2006) describe the steps required to isolate a fault and recover from it in the system.

As a preliminary result, we present two scenarios and the analysis of two associated procedures that illustrates how the procedures can be verified for branch coverage (metric 1b), and fewer branches (metric 2b).

In the first scenario, the observed symptom is “battery output voltage low anomaly”. The actual component that has failed in the system is the "Battery 1 voltage sensor, EI-135 in Figure 4. An excerpt from he procedure that deals with this anomaly is shown in Figure 5.

In summary, this procedure includes steps to:

1. *verify* the operational mode (or configuration) of the EPS system (in this case Battery 1 powers AC Load A1),
2. *check* the battery output voltage (EI 135 reading), and if low,
3. *command* Battery 1 *off* and Battery 2 *on*,
4. *command* Relay EY 241, EY 260, and EY 274 *closed,*
5. *check* the temperature of AC Load A2 (TE 505),
6. *verify* the reconfigured operational mode (or configuration) of the EPS system (now Battery 2 powers AC Load B2.)

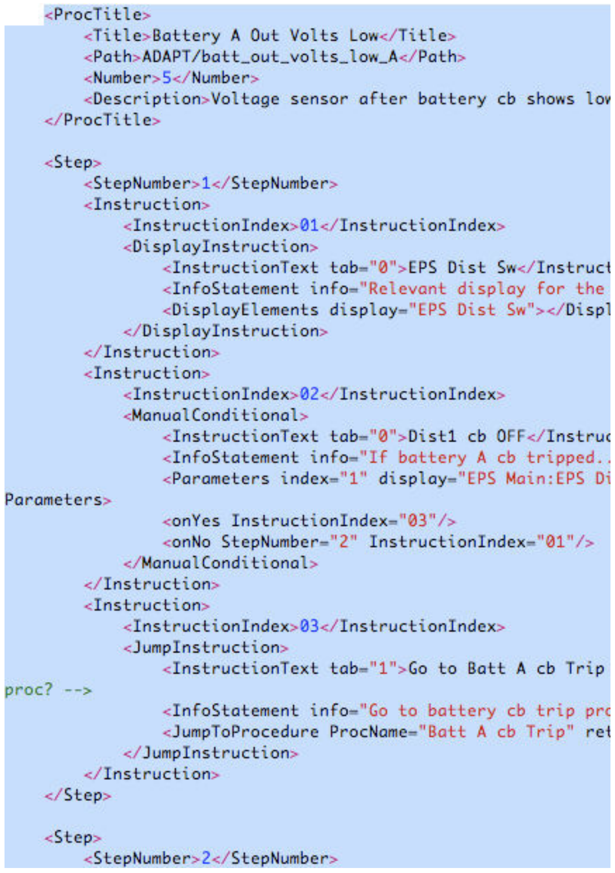
This procedure basically checks for a battery failure and reconfigures the system to use the redundant battery to power an identical load type, which was supported in the intended operational mode of the system.

However, the procedure is missing a "test" that could have disambiguated between a false alarm due to a sensor failure (EI 135) and an actual battery failure (Battery 1). As a result, it directly prompts for reconfiguration of the system to use the redundant battery power. The TEAMS model and the auto generated diagnostic tree can easily identify this "missing test" (metric 1b) which would eliminate the possibility of a sensor failure and verify an actual battery failure. The addition of this test to the procedures would have prevented an unnecessary and costly reconfiguration of the system.

In the second scenario, the observed symptom is "load bank relay position anomaly". The actual component that has failed in the system is the "Load Bank A Relay Position Sensor, ESH 170 in Figure 4.

In summary, the procedure that deals with this anomaly includes steps to:

1. *verify* the operational mode (or configuration) of the EPS system (in this case Battery 1 powers AC Load A1),
2. *verify* the relay position sensor output (ESH 170 reading) to be *open*,
3. *verify* Inverter 1 output voltage (EI 165) is within operational limits,
4. *if true, check* the temperature output of AC Load A1 (TE-500),
5. if within operational limits *conclude* ESH 170 sensor failure, or
6. if outside of operational limits *go to* Procedure Inverter 1 Output Voltage Anomaly,



```
    <ProcTitle>
        <Title>Battery A Out Volts Low</Title>
        <Path>ADAPT/batt_out_volts_low_A</Path>
        <Number>5</Number>
        <Description>Voltage sensor after battery cb shows lov
    </ProcTitle>

    <Step>
        <StepNumber>1</StepNumber>
        <Instruction>
            <InstructionIndex>01</InstructionIndex>
            <DisplayInstruction>
                <InstructionText tab="0">EPS Dist Sw</Instruct
                <InfoStatement info="Relevant display for the
                <DisplayElements display="EPS Dist Sw"></Displ
            </DisplayInstruction>
        </Instruction>
        <Instruction>
            <InstructionIndex>02</InstructionIndex>
            <ManualConditional>
                <InstructionText tab="0">Dist1 cb OFF</Instruc
                <InfoStatement info="If battery A cb tripped..
                <Parameters index="1" display="EPS Main:EPS Di
Parameters>
                <onYes InstructionIndex="03"/>
                <onNo StepNumber="2" InstructionIndex="01"/>
            </ManualConditional>
        </Instruction>
        <Instruction>
            <InstructionIndex>03</InstructionIndex>
            <JumpInstruction>
                <InstructionText tab="1">Go to Batt A cb Trip
proc? -->
                <InfoStatement info="Go to battery cb trip prc
                <JumpToProcedure ProcName="Batt A cb Trip" ret
            </JumpInstruction>
        </Instruction>
    </Step>

    <Step>
        <StepNumber>2</StepNumber>
```

Figure 5. Excerpt from a "battery output voltage low anomaly" procedure for the ADAPT EPS system.

7. if zero *conclude* EY 170 relay failure, or
8. *if false, go to ….*

This scenario basically checks for a load bank relay failure, disambiguates between a relay and relay sensor failure and reconfigures the system to use the redundant load bank in case of a relay failure (not included in the procedural steps above).

The procedure includes 3 checks (ESH 170, EI 165, and TE 500) to conclude that the anomaly is a relay sensor failure. However, the same diagnosis can be made by using only two of the available tests (ESH 170 followed by TE500). The TEAMS model and the auto generated diagnostic tree can easily identify this path with fewer tests (metric 2b), which would reduce the complexity of the procedure.

Running similar analyses using the DTV method, we found some critical ambiguity groups (of size three and four) in ADAPT. For example, there was no voltage sensor between a circuit breaker and a relay, so no procedural check can currently distinguish between failures of these two. In another case, the DTV method identified that a connector failure could not be distinguished from a sensor failure by available tests. In

addition, we found some faults that could only be detected by human observation of symptoms.
These preliminary results are promising in that they illustrate how the DTV method can be used by system designers, procedure developers, and flight operators in order to document gaps in procedures as well as to identify potential improvement areas.

6 DISCUSSION

There are several unique aspects of the developed DTV method.

First, it provides the ability to identify limitations, missing steps, and potential improvements that can be used for the verification of procedures. Using the DTV method, system designers can identify some adjustments to the procedures that might offer savings in terms of reduced complexity (e.g., fewer steps, fewer branches), increased efficiency (e.g., faster results), increased autonomy (e.g., more fault isolation done in software), and/or reduced cost (e.g., reduced usage of scarce resources).

The DTV approach is especially useful for complex systems with redundant elements or functional redundancy. As is illustrated by the second scenario, the diagnostic tree may show that the same fault space can be covered by checking only a certain subset of the available sensors. In this case, one can consider adjusting the procedure to take advantage of this improved efficiency option. This goes beyond traditional verification methods that focus solely on the correctness and consistency of the procedures.

Second, the DTV method provides flexibility in exploring alternative ways of performing diagnosis and recovery actions under changing operational conditions. For example, it can suggest appropriate next steps to operators that are not captured by the original procedure definitions. TEAMS also makes it easier to verify appropriate procedures where propagation of faults can occur in a large system with several functionally redundant units. For example, the diagnostic checks are divided into seven categories so that one can choose to produce diagnostic trees, for example, only involving checks of temperature sensors. Similarly, since some tests are appropriate only for certain modes of operation, one can restrict the diagnostic tree to those checks relevant to a particular mode of operation (e.g., when Battery 1 is powering Load 1).

Third, the DTV approach uses system models that are already being constructed by NASA as part of the development process. This means that design knowledge does not need to be captured twice. The DTV method enables the system designers to exploit existing models for verification of procedures.

Fourth, since the knowledge used by the DTV method is leveraged from existing diagnostic system models, it provides the ability to fuse information from multiple sensors and test points and to reason about multiple faults in a system for developing procedural steps that may fall outside the scope of original procedure definitions.

Fifth, being able to maintain the TEAMS model by adding new components and then producing an updated diagnostic tree reduces the risk that change introduces in the procedures.

Finally, challenges with which we are currently contending largely involve: (1) management of the relationships between the many-to-many elements in the model and in the procedural steps, (2) checks in the procedures involving human elements (e.g., intervention) not currently represented in the model and (3) scalability of the approach for larger systems. We hypothesize that our future work will show that Challenges 1 and 3 can be handled within the existing TEAMS framework and that Challenge 2 will encourage fuller representation of human-computer interactions in the models (i.e., will consider the human as part of the system to be modeled)

7 CONCLUSIONS AND FUTURE WORK

We presented a method, namely the Diagnostic Tree for Verification (DTV), developed with the goal of leveraging the information contained within auto-generated diagnostic trees in order to assist the verification of standard operating procedures. Preliminary results indicate that the method presented here facilitates the identification of potential gaps in the coverage of the standard operating procedures.

Specifically, our method helps identify: (1) inconsistencies between the information generated by diagnostic trees and operating procedures, (2) missing checks/tests and responses between diagnostic trees and operating procedures, (3) alternative paths for diagnosis/isolation and recovery actions than those suggested by operating procedures, and (4) an optimized strategy for diagnosis/isolation and recovery actions. With these unique capabilities, the DTV method offers opportunities for enhanced fault handling and increased mission safety and affectivity.

ACKNOWLEDGMENT

The research described in this paper was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, and Ames Research Center under a contract with the National Aeronautic and Space Administration and funded by NASA's OSMA Software Assurance Research Program.

REFERENCES

(Brat et al., 2008) Brat, G., M. Gherorghiu, D. Giannakopouluo, C. Pasareanu, "Verification of Plans and Procedures" in Proceedings of IEEE Aerospace Conference, 2008.

(Button and Chicatelli, 2005) Button R.M. and A. Chicatelli, "Electrical Power System Health Management", In Proc. 1st International Forum on Integrated System Health Engineering and Management in Aerospace, November 2005, Napa, CA.

(Damas et al., 2009) Damas C., B. Lambeau, F. Roucoux and Axel van Lamsweerde, "Analyzing Critical Process Models Through Behavior Model Synthesis", Proceedings of 31st International Conference on Software Engineering, May 16-24, 2009, Vancouver, Canada.

(Deb et al., 1995) Deb, S., Pattipati, K.R., Raghavan, V., Shakeri, M., Shrestha, R. "Multisignal flow graphs: a novel approach for system testability analysis and fault diagnosis", IEEE Aerospace and Electronics Systems Magazine, Vol.10, No. 5, pp. 14 -25, 1995.

(Frank, 2008). Frank G., "Automation for Operations", Proceedings of AIAA SPACE Conference and Exposition, September 9-11, 2008, San Diego, California.

(Ghosal and Azam, 2008) Ghosal S., and M. Azam, "Technology Transfer of Contingency Software Process", Phase III, Final Report, 2008, Qualtech Systems Inc.

(Kortenkamp et al., 2008) Kortenkamp, D., R. Peter Bonasso and D. Schreckenghost, "Developing and Executing Goal-Based, Adjustably Autonomous Procedures," in proceedings of the AIAA InfoTech@Aerospace Conference 2007.

(LTSA 2008) http://www.doc.ic.ac.uk/ltsa/eclipse/.

(Lutz and Patterson-Hine, 2008a) Lutz R. and A. Patterson-Hine, "Tool-Supported Verification of Contingency Design: Poster and Abstract", NASA SMD/PSD Fault Management Workshop, April 14 – 16, 2008, New Orleans, LA.

(Lutz and Patterson-Hine, 2008b) Lutz R., and A. Patterson-Hine, "Using Fault Modeling in Safety Cases", ISSRE 2008, pp. 271-276.

(Lutz et al., 2008) Lutz, R., A. Patterson-Hine, S. Nelson, C. Frost, D. Tal, and R. Harris, "Using Obstacle Analysis to Identify Contingency Requirements on an Unpiloted Aerial Vehicle", Requirements Engineering Journal, 12(1), Jan, 2007, pp. 41-54.

(Lutz et al., 2007) Lutz, R., A. Patterson-Hine, S. Poll, C. Domagala and S. Ghosal, "Tool-Supported Software Contingency Analysis," 1st International Workshop on Aerospace Software Engineering, in conjunction with 29th International Conference on Software Engineering (ICSE 2007), May 20-21, 2007, Minneapolis, MN.

(McCann et al., 2006) McCann, R., Beutter, B. R., Matessa, M., McCandless, J. W., Spirkovska, L., Liston, D., Hayashi,M., Ravinder, U., Elkins, S., Renema, F., Lawrence,R., & Hamilton, A. "Description and Evaluation of a Real-time Fault Management Concept for Next-generation Space Vehicles", 2006, Internal Report to Johnson Space Center.

(Patterson-Hine et al., 2005) Patterson-Hine, A., Narasimhan, S., Aaseng, G., Biswas, G., Pattipati, K., "A Review of Diagnostic Techniques for ISHM Applications." 1st Integrated Systems Health Engineering and Management Forum. Napa, CA. November 2005.

(Poll et al., 2007) Poll S., A. Patterson-Hine, J. Camisa, D. Garcia, D. Hall, C. Lee, O. J. Mengshoel, C. Neukom, D. Nishikawa, J. Ossenfort, A. Sweet, S. Yentus, I. Roychoudhury, M. Daigle, G. Biswas, and X. Koutsoukos, "Advanced Diagnostics and Prognostics Testbed", In Proc. of the 18th International Workshop on Principles of Diagnosis (DX-07), Nashville, TN, May 2007.

(Polyspace 2008) http://www.polyspace.com

(QSI 2009) QSI, Testability Engineering and Maintenance System (TEAMS) Tool, www.teamsqsi.com.

(Verma et al., 2005). Verma V., T. Estlin, A. Jónsson, C. Pasareanu, R. Simmons, K. Tso, "Plan Execution Interchange Language (PLEXIL) for Executable Plans and Command Sequences", International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS), 2005.

(Visser et al., 2003) Visser W., K. Havelund, G. Brat, S. Park and F. Lerda. "Model Checking Programs." In the Automated Software Engineering Journal, Vol. 10, number 2, April 2003.

Tolga Kurtoglu is a Research Scientist with Mission Critical Technologies at the Intelligent Systems Division of the NASA Ames Research Center working for the Systems Health Management group. His research focuses on the development of prognostic and management systems, model-based diagnosis, design automation and optimization, and risk and reliability based design. He received his Ph.D. in Mechanical Engineering from the University of Texas at Austin in 2007 and has an M.S. degree in the same field from Carnegie Mellon University. Dr. Kurtoglu has published over 40 articles and papers in various journals and conferences and is an active member of ASME, ASEE, AIAA, and AAAI. Prior to his work with NASA, he worked as a professional design engineer at Dell Corporation in Austin, Texas.

Robyn R. Lutz received the Ph.D. degree from the University of Kansas (1980). She has worked at Jet Propulsion Laboratory/Caltech, since 1983, currently as a senior engineer in the Flight Software and Data Systems section. She is also a professor in the Department of Computer Science at Iowa State University. Her work focuses on software safety, software product lines, defect analysis, and formal modeling and analysis, especially for fault detection and recovery. Her research is supported by NASA and the National Science Foundation. She is a member of ACM and a Senior Member of IEEE.

Ann Patterson-Hine received her Ph.D. in 1988 in Mechanical Engineering from the Univ. of Texas at Austin. She is Tech Area Lead for Discovery and Systems Health in the Intelligent Systems Division at NASA Ames Research Center. She is the Principal Investigator for Integrated Systems Health Management under NASA's Exploration Technology Development Program. Her research has centered on the use of engineering models for model-based reasoning in advanced monitoring and diagnostic systems. She is a member of AIAA and a Senior Member of IEEE.