Using Model-based Assurance to Strengthen Diagnostic Procedures

Robyn Lutz
Jet Propulsion Lab, California Institute of Technology
& Iowa State University
robyn.r.lutz@jpl.nasa.gov

Jeremy Johnson
SGT/NASA Ames Research Center
Moffett Field, CA USA
jeremy.r.johnson@nasa.gov

Ann Patterson-Hine
NASA Ames Research Center
Moffett Field, CA USA
ann.patterson-hine@nasa.gov

***Abstract*—In previous work we described Diagnostic Tree for Verification (DTV), a partially automated software engineering technique by which diagnostic trees generated from system models are used to help check out diagnostic procedures. Diagnostic procedures are instructions used to isolate failures during operations. Assuring such procedures manually is time-consuming and costly. This paper reports our recent experience in applying DTV to diagnostic procedures for lighting failures in NASA's Habitat Demonstration Unit (HDU), a prototype for astronauts' living quarters. DTV identified missing and inconsistent instructions, as well as more-efficient sequences of diagnostic steps. Unexpectedly, the most significant benefit was finding assumptions that will not remain true as the system evolves. We describe both the challenges faced in applying DTV and how its independent perspective helped in assuring the procedures' adequacy and quality. Finally, the paper discusses more generally how software systems that are model-based, rapidly evolving and safety-critical appear most likely to benefit from this approach.**

Keywords-diagnostic procedures; model-based; trouble-shooting; automated analysis

I. INTRODUCTION

This experience paper describes our application of a commercial automated software engineering toolset to a NASA system to analyze its diagnostic procedures. Diagnostic procedures are step-by-step instructions to help users troubleshoot what is wrong in an operational system when a failure occurs. Assuring the correctness and completeness of diagnostic procedures manually, as is done now, is time-consuming and costly because it depends heavily on review by domain experts [3].

We report recent experience analyzing the lighting system diagnostic procedures for NASA's Habitat Demonstration Unit (HDU) [6]. The HDU is a prototype of sustainable living quarters, workspaces and laboratories for next-generation space missions (Fig. 1). The HDU diagnostic lighting procedures are safety-critical. The contingency scenario of a crew experiencing loss of interior lighting requires accurate troubleshooting to restore lighting promptly.

In previous work we described Diagnostic Tree for Verification (DTV), a technique in which diagnostic trees automatically generated from system models are used to verify diagnostic procedures [4]. The contribution of this paper is that our experience shows that these auto-generated trouble-shooting trees are an effective way to assure the correctness of diagnostic procedures and an improvement over the current practice of manually reviewing diagnostic procedures. The use of DTV identified missing steps in the procedures, missing failure sources, inconsistent instructions, and undocumented assumptions. It also found better, alternative sequences of diagnostic steps. Unexpectedly, the most significant benefit was finding assumptions that will not remain true as the system evolves.

The rest of the paper is organized as follows. Section II presents the application. Section III describes the technique used. Section IV discusses the results and lessons learned for practitioners in terms of other applications and for researchers in terms of future work. Section V notes related work. Section VI is a brief conclusion. The results suggest that broader use of trouble-shooting trees automatically generated from system failure models can provide an independent perspective and reduce the manual effort of assuring the adequacy and quality of diagnostic procedures.

II. APPLICATION

There were three main application inputs to our analysis. In addition, we learned from the visit of the second author to the actual HDU at Johnson Space Center, and from domain experts' answers to our questions.

Figure 1. Habitat Demonstration Unit (HDU) [Courtesy NASA]

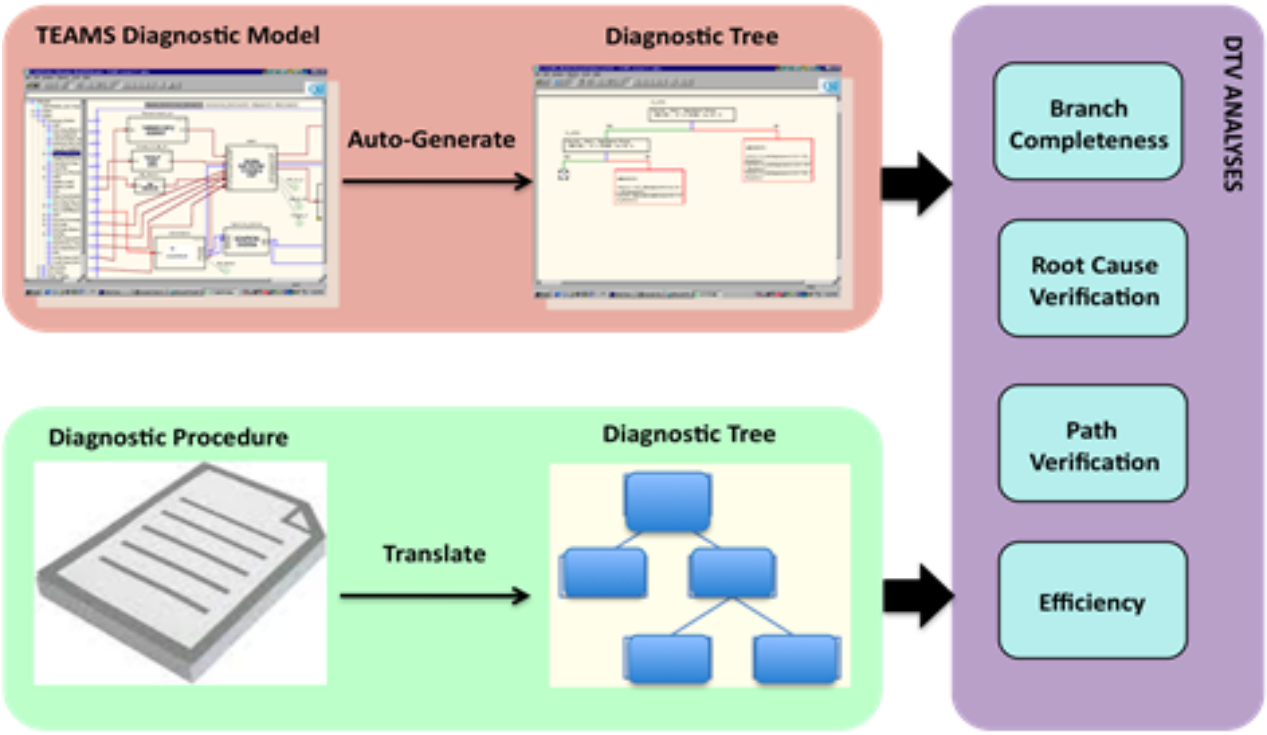

Figure 2. Overview of DTV technique

Operational handbook. The HDU project's 300+ page handbook documents the normal operation of the HDU in structured text. It provides a goal-based view of operations. For example, one section describes the steps to be taken in order to activate the lighting in the entryway, while another describes the steps to de-activate the lighting. The handbook gives details of the architecture of the HDU which were useful to us in understanding the system. Consideration of abnormal situations such as failure conditions is not part of the operational handbook.

Diagnostic procedures. The HDU project described the procedures to diagnose problems during operations for the HDU lighting system in a graphical format, with labeled boxes showing steps (e.g., "check that x is on") , each with two outbound arrows (check passed/check failed). Some steps are checks done by the avionics software (e.g., of switch settings), while others are checks done by the astronauts at the crew display, visual inspections (is the LED lit?), or manipulations (is the dimmer turned to low?).

Diagnostic model. The HDU model represents the system's failure effects, how failures propagate through the system, and the observable conditions (symptoms) those failures manifest. The model already had been built using the commercial TEAMS toolset [8]. The model has 226 failure modes and 203 tests. It is based on a block diagram of the HDU, and includes both software and hardware modules. Test points represent the physical or computational locations of checks using sensor data or other means to observe the system. As always, results depend on the quality of the model. We verified the model against the HDU schematics and the proposed telemetry file, giving us some confidence in its accuracy.

TEAMS automatically produces a diagnostic tree from the model showing an optimized (based on user settings) branching sequence of checks in order to troubleshoot a symptom during operation. We used these trees to evaluate the diagnostic procedures.

III. DTV TECHNIQUE

This section describes the DTV technique that we applied. DTV has five steps [4]:

1. The system's failures are formally modeled in a diagnostic model, as described above.
2. A set of diagnostic procedures (here, the HDU lighting system) is selected for DTV analysis.
3. Symptoms that trigger the selected procedures are added to the model. We added these symptoms to the existing model as part of our analysis.
4. A set of diagnostic trees is auto-generated from the model for each symptom point in the model.
5. Analyses are conducted by comparing the elements and steps in the procedures with those in the diagnostic trees generated from the model.

Fig. 2 gives an overview of the DTV technique. At the top left is the TEAMS model and to the right of it the diagnostic tree auto-generated from it. In the bottom left is the text-based diagnostic procedure and to the right of it the hand-generated tree representation from it. Using the HDU project's box-and-lines procedural representation, we manually converted the steps in each procedure to a tree representation similar in style to the trees that TEAMS auto-generates to aid with the comparison analyses.

The right-hand side of Fig. 2 shows the four analyses we conducted using tree-to-tree comparisons. The associated findings are described further in the next section.

Branch Completeness Analysis to identify inconsistencies in the trees' structures. This confirms that every non-leaf node in the textual procedure's diagnostic tree has two outgoing branches, i.e., a passed and a failed check, that each branch leads either to a diagnosis or another check, that there are no duplicate or negated tests, and that the successor nodes are the same. A difference we found is that the model-based diagnostic tree checks the crew display (an automated test) before manually checking the dimmer settings. We recommended an alternative sequencing.

Root Cause Coverage Analysis to verify that all potential root causes of failure are included in the procedure. This confirms that a symptom's tree's leaf nodes include all the possible causes of the symptom, via comparison of the two diagnostic trees in the context of single component failures, and that for any leaf node consisting of a set of several failures (called an "ambiguity group"), it is not possible to isolate the failure further. We found failures in the model-generated trees that were not included in the procedures, e.g., a check of the heat pump. One model-based tree included an ambiguity group that was not in the procedure. This was because the additional failures were represented in the model-based tree, giving it a more detailed diagnosis of power-system causes of lighting failures.

Path Verification Analysis to verify that a path in a procedure results in the correct diagnosis. We take a path from the procedure's diagnostic tree for a symptom, and input to TEAMS the symptom itself (as a "failed" test), along with each of the outcomes along the path (according to whether each is a "passed" or "failed" test). TEAMS calculates the status of each component, and we confirm that the procedure's path's diagnosis agrees with the TEAMS-calculated status of components.

Efficiency Analysis to identify whether the same set of root causes can be isolated with fewer tests by comparing the two diagnostic trees. We skip this here due to limited space and no interesting results.

IV. Results

We first describe the findings from the DTV analysis of the HDU troubleshooting procedures and then consider the broader implications regarding assurance of diagnostic procedures for practitioners and research.

A. Findings for the HDU

Incompleteness:

- Missing steps in the procedure. In some cases the procedure did not check for faults that could cause a lighting failure. An example is that the lighting software remote interface unit is not checked in any procedure.
- Missing failure sources. In some cases, the auto-generated tree checked for failed components that were not considered in the troubleshooting procedures. This is because the model contained additional components that could cause the lighting failure and that were not considered during the development of the troubleshooting procedures. An example is a junction box failure that would cut power to the lights.
- Missing procedure. There were procedures for the failure of a single light and of all lights, but not for a common-cause failure for some but not all lights failing. This procedure would be different from the all-lights out procedure because some checks become unnecessary (infeasible if any light comes on) and other checks become feasible since there is some lighting available.

Usability:

- Inconsistent usage. In one troubleshooting procedure "yes" sometimes meant "passed" and sometimes meant "failed." Similarly, one procedure checked that the light was ON while another checked that the light was OFF.
- Inconsistent level of detail. While some steps were described precisely, others were described too imprecisely to be useful, e.g., verify that the resources "work properly".

Alternative Sequences of steps:

- More efficient sequencing. The model-based diagnostic tree pushed time-consuming manual tests to the end of the sequence of checks, while the procedures did not. For example, the diagnostic procedure checks each switch, which is time-consuming, before checking that external power is available, while the diagnostic tree checks the external power first, a single check.

Assumptions:

- Undocumented or missing assumptions. The DTV analysis found, e.g., that a procedure implicitly assumes that everything upstream of the crew display works, while the diagnostic model includes those tests. As a result, the model-based diagnostic tree had more failures and more tests in it than the procedural tree.
- Invalid assumptions. Some assumptions found during the DTV analysis seemed unrealistic, e.g., some timing deadlines were too restrictive to be practical in the dark. There was also an implicit assumption that the lights are all on the same circuit. This is true in the current architecture (and model), but unlikely to remain true as the HDU evolves. For example, an "attic" will soon be added. Such changes are readily incorporated into the model, which is updated regularly, but the consequences for diagnostic procedures would be easy to miss as the system changes, with potentially hazardous effects.

Not surprisingly, the comparison between the two diagnostic trees also found a few missing elements in the model, such as the Power Distribution Unit ports. The model was updated to incorporate these and the diagnostic trees re-run. This side effect of improving the model gives increased confidence in the model's (and hence the diagnostic tree's) completeness.

B. Lessons Learned for Practitioners

When to use failure models to evaluate diagnostic procedures? Our work leveraged an existing TEAMS HDU model to provide an independent assessment of the HDU's diagnostic procedures. This effort was largely successful. Applying the four types of DTV analysis to compare the trees gave a structured and fairly easy way to identify mismatches

between the two. The comparison caught multiple issues of interest to the project that will improve troubleshooting procedures for operations. DTV seems especially well-suited to rapidly evolving systems and to systems where diagnostic procedures are safety-critical. It is scalable in that operational procedures usually have a relatively small number of steps, and the size of the model-based trees can be controlled by user-set parameters.

What if a TEAMS model does not exist? We speculate that the DTV approach can be adapted to work with any machine-readable failure model, and have proposed work to investigate this further next year. On the other hand, for projects where no machine-readable failure model exists, then building one just to check out the diagnostic operating procedures is impractical. Thus, a precondition for use of DTV is the existence of a model that represents backward analysis of failure root causes from symptoms.

What does a model-based approach contribute over expert review? The model and the procedures were, as is typical of most organizations, developed by different groups. Thus, the model-based, trouble-shooting tree and the tree representation of the procedure approached the diagnosis and isolation of operational failures from slightly different perspectives. We found and documented gaps that had not been identified by informal reviews of the procedures.

C. Lessons Learned for Research

There are several challenges that currently limit what we want to represent and evaluate.

Cost function. A consistent cost function is needed to evaluate tradeoffs between alternatives that may exchange more tests for more root-cause isolation. It should take into account the likelihood of the failure cause and the level of effort (e.g., automated, visual inspection, manipulation) required for each diagnostic step. TEAMS has the capability to incorporate such a function into the model.

Situation-based sequencing. In some cases the preferred sequencing of diagnostic steps depends on the situation. For example, if an astronaut is alone in the HDU then the best sequence of diagnostic steps may be different from the best steps if there are several astronauts who can share the tasks. The diagnostic tree can represent modes.

Reconfiguration to isolate faults. Often the ability to reconfigure the system during the troubleshooting process can reap valuable knowledge. Representations of system configuration were not included in either models or the textual procedures. The work reported here, which used the TEAMS Designer and TEAMS/RT did not include reconfiguration to isolate faults. However, another member of the tool suite, TEAMMATE, has that capability, so we plan to incorporate reconfiguration to isolate faults.

V. Related Work

Although manual inspection and review are still the most widely used approaches for assuring procedures in practice, there have been several recent advances in automated techniques to help develop and verify procedures. Some authoring tools support syntax checking [3], while static analyzers can verify the syntax and well-formedness of procedures [1]. Machine-readable representations of procedures have been developed [9], and work is underway to generate procedures automatically from models.

Model checkers have been used to assure that processes satisfy essential properties, most notably in the medical field [2]. The focus is on assembling process fragments into a model for automated verification rather than, as here, on using existing models to check operational procedures. Our work is also related to work in automated diagnostic software for monitoring and diagnosis of dynamical systems [7].

VI. Conclusion

Our experience with the HDU shows that comparisons with auto-generated trouble-shooting trees are an effective way to improve the review of diagnostic procedures for operations.

Acknowledgment

This research was carried out at NASA Ames Research Center and at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The work was sponsored by the NASA Office of Safety and Mission Assurance under the Software Assurance Research Program led by the NASA Software IV&V Facility. This activity is managed locally at JPL through the Assurance and Technology Program Office.

References

[1] G. Brat, M. Gherorghiu, D. Giannakopouluo, C. Pasareanu, "Verification of Plans and Procedures" Proc. IEEE Aerospace Conference, 2008.

[2] C. Damas, B. Lambeau, F.Roucoux and Axel van Lamsweerde, "Analyzing Critical Process Models Through Behavior Model Synthesis", Proc. 31st Int'l Conf on Software Engineering (ICSE), 2009.

[3] D. Kortenkamp, R. Peter Bonasso and D. Schreckenghost, "Developing and Executing Goal-Based, Adjustably Autonomous Procedures," Proc. AIAA InfoTech@Aerospace Conference, 2007.

[4] T. Kurtoglu, R. Lutz and M. Feather, "Model-Based Assurance of Diagnostic Procedures for Complex Systems", Proc. Annual Conf. of Prognostics and Health Management Society (PHM), 2010.

[5] R. Lutz, 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.

[6] NASA Habitat Demonstration Unit Project http://www.nasa.gov/exploration/analogs/hdu_project.html

[7] A. Patterson-Hine, A., G. Aaseng, G. Biswas, S. Narasimhan and K. Pattipati, "A Review of Diagnostic Techniques for ISHM Applications." Proc. 1st Int'l Forum on Integrated System Health Engineering and Management (ISHEM), 2005.

[8] QSI, Testability Engineering and Maintenance System (TEAMS) Tool, www.teamsqsi.com.

[9] V. Verma V., T. Estlin, A. Jónsson, C. Pasareanu, R. Simmons, K. Tso, "Plan Execution Interchange Language (PLEXIL) for Executable Plans and Command Sequences," Proc. 8th Int'l Symp on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), 2005.