Systematic Benchmarking of Diagnostic Technologies for an Electrical Power System

Tolga Kurtoglu
Mission Critical Technologies
NASA Ames Research Center
Moffett Field, CA 94035
650-604-1738
tolga.kurtoglu@nasa.gov

David Jensen
Complex Engineered Systems Design Laboratory
Oregon State University, Corvallis, OR 97331

Scott Poll
NASA Ames Research Center
Moffett Field, CA 94035

Abstract—Automated health management is a critical functionality for complex aerospace systems. A wide variety of diagnostic algorithms have been developed to address this technical challenge. Unfortunately, the lack of support to perform large-scale V&V (verification and validation) of diagnostic technologies continues to create barriers to effective development and deployment of such algorithms for aerospace vehicles. In this paper, we describe a formal framework developed for benchmarking of diagnostic technologies. The diagnosed system is the Advanced Diagnostics and Prognostics Testbed (ADAPT), a real-world electrical power system (EPS), developed and maintained at the NASA Ames Research Center. The benchmarking approach provides a systematic, empirical basis to the testing of diagnostic software and is used to provide performance assessment for different diagnostic algorithms.[1 2]

TABLE OF CONTENTS

1. INTRODUCTION 1
2. ADVANCED DIAGNOSTICS AND PROGNOSTICS TESTBED 2
3. BENCHMARKING ARCHITECTURE 3
4. TEST SCENARIOS 3
5. EXPERIMENTAL PROTOCOL 4
6. EVALUATION METRICS 5
7. DIAGNOSTIC ALGORITHMS 6
8. PERFORMANCE ASSESSMENT 6
9. CONCLUSIONS 7
10. ACKNOWLEDGMENT 7
REFERENCES 8
BIOGRAPHY 8

1. INTRODUCTION

The complexity of some modern systems, especially those related to the aerospace field, has motivated the development of automated health management technologies. NASA's spacecraft and aircraft contain multiple sub-systems including navigation systems, power systems, and propulsion systems, and it is essential to keep these systems healthy during a mission.

A crucial enabling technology to address this need is *diagnostic algorithms*. These algorithms are designed to detect and isolate anomalies of either a component or the whole system based on observations received from a complex system. In recent years, a wide range of algorithms, both model-based and data-driven, have been developed to increase autonomy and improve system reliability and affordability.

Unfortunately, the lack of support for comparative analysis of different diagnostic algorithms still continues to create barriers to effective development and deployment of such algorithms for aerospace vehicles. This makes it difficult to understand the pros and cons of different alternatives, which might lead to sub-optimal design choices being made, with obvious unfortunate consequences for system performance and safety.

Moreover, there is a lack of systematic approaches to perform large-scale V&V (verification and validation) of aerospace software - including diagnostic technologies - that is closely aligned with V&V and reliability activities as they relate to hardware. This difficulty hinders the transition of techniques and algorithms from state of the art into the state of practice.

In this paper, we describe an architecture and a formal framework developed for benchmarking of diagnostic

[1] 978-1-4244-2622-5/09/$25.00 ©2009 IEEE
[2] IEEEAC paper #1225, Version 3, Updated Nov 3, 2008

technologies. The diagnosed system is the Advanced Diagnostics and Prognostics Testbed (ADAPT), a real-world electrical power system (EPS), developed and maintained at the NASA Ames Research Center. The benchmarking approach provides a systematic, realistic approach to perform V&V, in the form of testing, of diagnostic software. In particular, the approach is suitable for systems that combine hardware (like EPS) and software (like EPS diagnosis). While in this work we use the ADAPT EPS, other integrated hardware and software systems can be treated in a similar fashion.

2. ADVANCED DIAGNOSTICS AND PROGNOSTICS TESTBED

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 [1,2], ADAPT provides a representative aerospace vehicle EPS that enables automated diagnosis in a complex domain. The main functions and layout of the ADAPT power system is shown in Fig. 1. The EPS can deliver AC (Alternating Current) and DC (Direct Current) power to loads, which in an aerospace vehicle would include subsystems such as the avionics, propulsion, life support, and thermal management systems.

ADAPT contains elements common to many aerospace applications: *power generation*, *power storage*, and *power distribution*. Three power generation sources are connected to three sets of batteries, which in turn supply two load banks. Each load bank has provisions for 6 AC loads and 2 DC loads.

The two sources of *power generation* include two battery chargers. The battery chargers are connected to appropriate wall outlets through relays. Hardware relay logic prevents connecting one charge source to more than one battery at the same time, and from connecting one charging circuit to another charging circuit.

The *power storage* consists of three battery modules, which are used to store energy for operation of the loads. Each "battery" consists of two 12-volt sealed lead acid batteries connected in series to produce a 24-volt output. Two battery sets are rated at 100 amp-hrs and the third set is rated at 50 amp-hrs.

Any of the three batteries can be used to power any 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 AC and DC loads. All relays are of the normally-open type. An inverter converts the 24-volt DC battery input to a 120-volt rms 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,

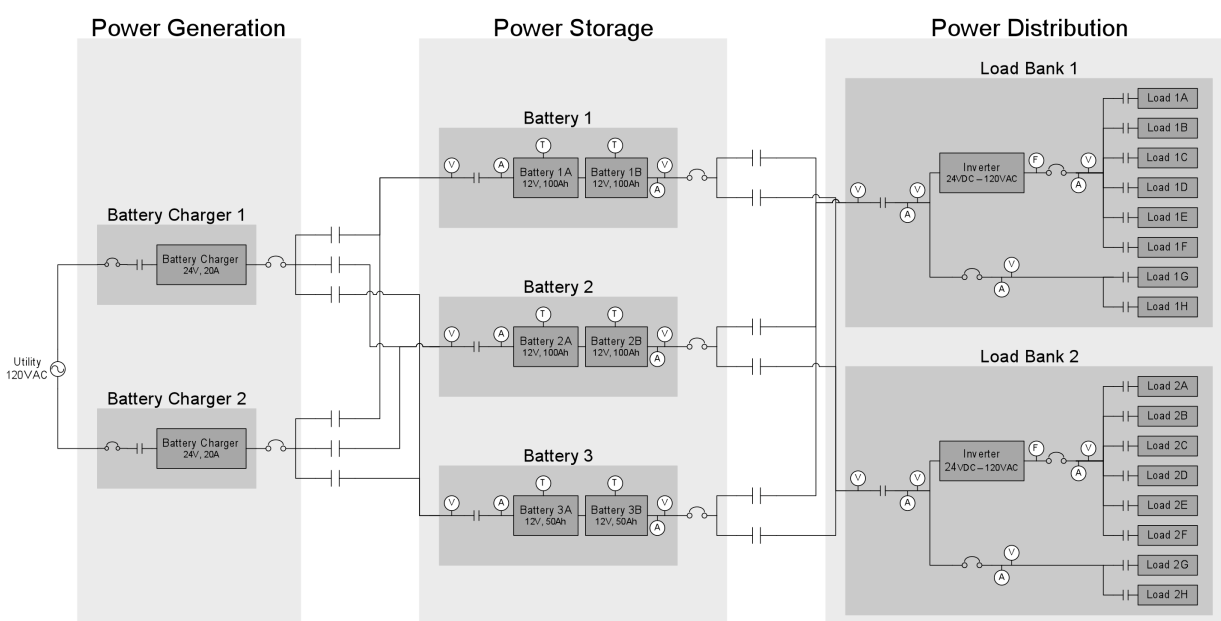

Fig. 1. ADAPT Testbed components and the system configuration

switch positions, light intensities, and AC frequencies, and includes over 100 sensors. (More information on the ADAPT testbed can be found in [2].)

3. BENCHMARKING ARCHITECTURE

Several researchers have attempted to demonstrate benchmarking capability on different systems [3-5]. Among these, Orsagh et al. [3] provided a set of 14 metrics to measure the performance and effectiveness of prognostics and health management algorithms for US Navy applications [4]. Bartys et al. [5] presented a benchmarking study for actuator fault detection and identification (FDI). This study, developed by the DAMADICS Research Training Network, introduced a set of 18 performance indices used for benchmarking FDI algorithms on an industrial valve-actuator system. Simon, et al. [6] introduced a benchmarking technique for gas path diagnostic methods to assess the performance of engine health management technologies.

The benchmarking methodology presented here extends prior work [7-10, 3-5] by defining a number of novel benchmarking indices [11], by providing a generic, application independent architecture that can be used for different monitoring and diagnostic algorithms, and by facilitating the use of real process data on a large-scale, complex engineering system.

The detailed architecture of the developed framework is presented in Fig. 2. The physical system is the Advanced Diagnostics and Prognostics Testbed (ADAPT). The EPS problem domain contains both discrete and continuous faulty behavior, which are defined under a standardized fault catalog. The fault catalog establishes a common baseline as to what failure modes and faulty behavior are required to be modeled by individual diagnostic algorithm developers.

The experimental testing procedure is usually scenario-based, where each scenario may have single or multiple

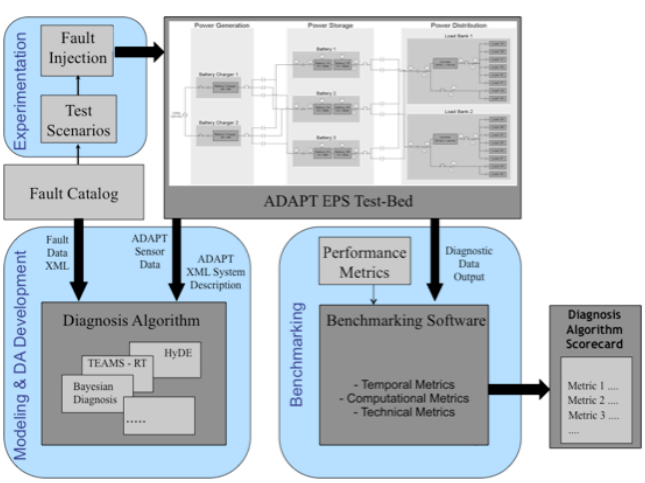

Fig. 2. The benchmarking architecture

faults injected into the system. This is unlike other benchmarking methods [3-6], which are mostly restricted to single fault assumption. To detect faults, each diagnostic algorithm has access to real-time data from the ADAPT EPS. Moreover, a standardized output scheme is enforced on the diagnostic algorithms to ensure the generation of common data sets for the calculation of metrics. The data from the testbed and this output of the diagnostic system are saved to a database, and the diagnostic algorithm performance is evaluated according to a predefined set of metrics.

4. TEST SCENARIOS

The ADAPT testbed offers a number of challenges to health management applications. The electrical power system shown in Figure 1 is a hybrid system with multiple system configurations made possible by switching among the generation, storage, and distribution units. Timing considerations and transient behavior must be taken into account when designing diagnosis algorithms. When power is input to the inverter there is a delay of a few seconds before power is available at the output. For some loads, there is a large current transient when the device is turned on. System voltages and currents depend on the loads attached, and noise in the sensor data becomes more pronounced as more loads are added [2]. Due to the low probabilities of failure, injection of faults is needed. Using the testbed software architecture, it is possible to inject multiple faults into the system.

ADAPT supports the repeatable injection of faults into the system in one of three ways:

Hardware-Induced Faults: These faults are physically injected at the testbed hardware. A simple example is tripping a circuit breaker using the manual throw bars. Another is using the power toggle switch to turn off the inverter. Relays may be failed by short-circuiting the appropriate relay terminals. Faults may also be introduced in the loads attached to the EPS. For example, the valve can be closed slightly to vary the back pressure on the pump and reduce the flow rate.

Software-Induced Faults: In addition to fault injection through hardware, faults may be introduced via software. Software fault injection includes one or more of the following: 1) sending commands to the testbed that were not intended for nominal operations; 2) blocking commands sent to the testbed; and 3) altering the testbed sensor data. The sensor data can be altered in a number of ways, as illustrated in Figure 3. For a static fault, the data are frozen at previous values and remain fixed. An abrupt fault applies a constant offset to the true data value. An incipient fault applies an offset that starts at zero and grows linearly with time. Excess sensor noise is introduced by adding Gaussian or uniform noise to the measured value. It is also possible to inject intermittent data faults, data spikes, and to combine more than one fault type for a given sensor at the same time.

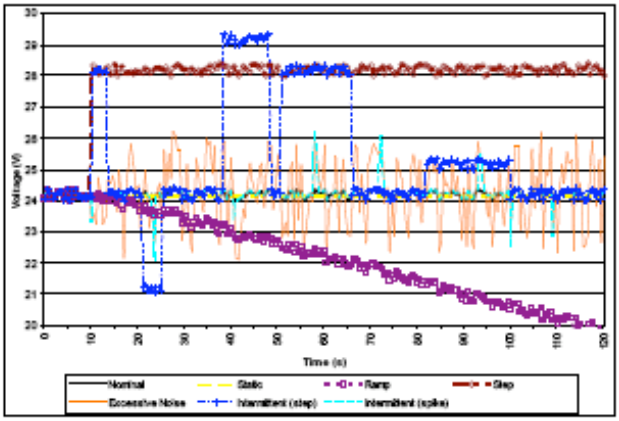

Fig. 3. Example fault types [2]

By using these approaches to software fault injection, fault scenarios may be constructed that represent a diverse range of sensor faults.

Real Faults: In addition the aforementioned two methods, real faults may be injected into the system by using actual faulty components. Simple examples are a blown light bulb or a degraded battery.

Since some fault scenarios may be costly, dangerous, or impossible to introduce in the actual ADAPT hardware, a simulation module called VIRTUAL ADAPT also provides fault injection capabilities. For example, degradation in the batteries can be simulated as an incipient change in a battery capacitance parameter. Other parametric faults can also be injected and simulated. In addition, VIRTUAL ADAPT permits experimentation with fault scenarios that cannot be realized in the hardware, such as an inverter malfunction. Currently, mostly discrete failures (e.g., relay failures) and sensor errors are introduced into ADAPT, so the simulation provides added functionality by enabling injection of other types of fault scenarios.

The faults that are injected to the testbed are listed in Table 1.

Table 1. Faults injected into the ADAPT testbed

Component	Fault Description
Battery	
	Degraded
Boolean Sensor	
	Stuck at Value
	Random Value
Circuit Breaker	
	Tripped
	Failed Open
	Stuck Closed
Inverter	
	Failed Off
Relay	
	Stuck Open
	Stuck Closed
Sensor	
	Random Value
	Drift Slope
	Stuck Value
	Offset
	Noise
Pump (Load)	
	Flow Blocked
	Failed Off
Fan (Load)	
	Over Speed
	Under Speed
	Failed Off
Light Bulb (Load)	
	Failed Off

5. EXPERIMENTAL PROTOCOL

The algorithms are tested against a number of diagnostic scenarios. Each diagnostic session defines some standard key points and intervals, which are best illustrated by Figure 4. Figure 4 splits the diagnostic session into three time intervals: $\Delta_{startup}$, $\Delta_{injection}$, $\Delta_{shutdown}$. During the first interval, the diagnostic algorithm is given time to initialize, read data files, etc. Though sensor observations may be available during $\Delta_{startup}$, no faults will be injected at this time. Fault injection and diagnosis takes place during $\Delta_{injection}$. Finally, to promote good programming practices, the algorithms are given some time to gracefully terminate during $\Delta_{shutdown}$. After this time, live diagnostic processes are killed and the system is recycled for the next diagnostic experiment.

Below are some notable points for the example diagnostic scenario from Fig. 4:

t_{inj} - A fault is injected at this time;

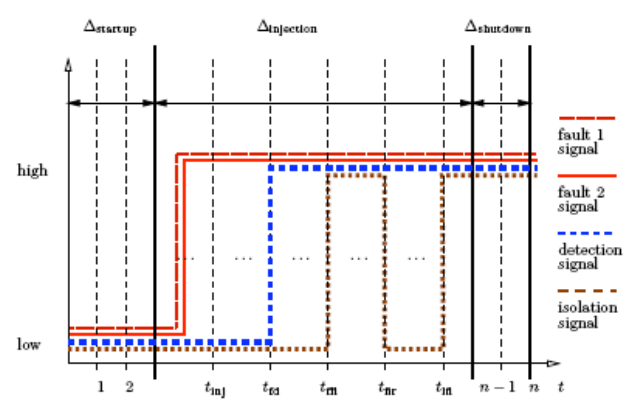

Fig. 4. Key time points, intervals, and signals

t_{fd} - The diagnostic algorithm has detected a fault;

t_{ffi} - The diagnostic algorithm has isolated a fault for the first time;

t_{lfi} - The diagnostic algorithm has converged to an isolated fault;

At the end of the diagnostic session the *scenario recorder* collects the following time-series and (actual and hypothesized) fault data to be used in the metrics computation:

T_{inj} - Fault injection signal;

T_{fd} - Fault detection signal;

ω^* - An actual fault set (once all faults are injected);

$\Omega = \{\varpi_1, \varpi_2, \ldots, \varpi_n\}$ - A (possibly empty) set of candidate diagnoses.

6. Evaluation Metrics

A set of thirteen metrics has been defined for assessing the performance of the diagnostic algorithms. These are listed in Table 2. (Detailed definition of the metrics can be found in [11].)

Table 2. Diagnostic Performance Metrics

Type	Diagnostic Objectives	Performance Metric
Detect	Time	Time to Detect
	Accuracy	Detection False Positive Rate
	Accuracy	Detection False Negative Rate
	Accuracy	Fault Detection Rate
	Accuracy	Fault Detection Accuracy
	Sensitivity	Detection Sensitivity Factor
	Stability	Detection Stability Factor
Isolate	Response	Time to Isolate
	Computation	Time to Estimate
	Accuracy	Isolation Classification Rate
	Accuracy	Isolation Misclassification Rate
	Resolution	Size of Isolation Set
	Stability	Isolation Stability Factor

In this paper, we focus on 6 of these metrics. Table 3 provides a summary of the notation used throughout this section. These 6 metrics are defined as:

Table 3. Metrics Notation

Symbol	Description
S	The set of scenarios for a given system description.
S_n	The set of nominal scenarios for a given system description.
S_f	The set of faulty scenarios for a given system description.
t_{fd}	The time when the fault detection signal has been asserted for the first time.
t_{inj}	The time when the first fault has been injected.
t_{fi}	The time when the fault isolation signal has been asserted for the last time.
ω^*	The injected fault.
Ω	The last set of faults returned by the diagnostic engine.
(ω, c)	The actual health state of a component c in a diagnosis ω.

Time to Detect (M_{TD}): The reaction time for a diagnostic engine in detecting an anomaly [11].

$$M_{TD} = t_{fd} \tag{1}$$

Time to Isolate (M_{TI}): The time for isolating a fault [11]. In many applications this metric is less important than the diagnostic accuracy, but it is important in sequential diagnosis, probing, etc.

$$M_{TI} = t_{lfi} \tag{2}$$

where t_{lfi} is the first instance of time when the latest persistent fault isolation signal has been issued.

Detection False Positive Rate (M_{FPR}): The metric that penalizes diagnostic algorithms which announce spurious faults [11]. The false positive rate is defined as:

$$M_{FPR} = \frac{\sum_{s \in S} m_{fp}(s)}{|S|} \tag{3}$$

where for each scenario s the "false positive" function $m_{fp}(s)$ is defined as:

$$m_{fp}(s) = \begin{cases} 1, \text{ if } t_{fd} < t_{inj} \\ 0, \text{ otherwise} \end{cases} \tag{4}$$

$$\text{where } t_{inj} = \infty \text{ for a nominal scenario}$$

Detection False Negative Rate (M_{FNR}): The metric that measures the ratio of missed faults by a diagnostic algorithm [11].

$$M_{FNR} = \frac{\sum_{s \in S_f} m_{fn}(s)}{|S_f|} \tag{5}$$

where for each scenario s the "false negative" function $m_{fn}(s)$ is defined as:

$$m_{fn}(s) = \begin{cases} 1, \text{ if } t_{fd} = \infty \\ 0, \text{ otherwise} \end{cases} \quad (6)$$

Fault Detection Accuracy (M_{FDA}): The fault detection accuracy is the ratio of number of correctly classified cases to the total number of cases [11]. It is defined as:

$$M_{FDA} = 1 - \frac{\sum_{s \in S} m_{fp}(s) + m_{fn}(s)}{|S|} \quad (7)$$

Isolation Classification Rate (M_{ICR}): Isolation classification rate metric measures the accuracy of the fault isolation by a diagnostic algorithm [11]. Let us denote the injected fault as ϖ^*. The isolation classification rate is then defined as:

$$M_{ICR} = \sum_{c \in |COMPS|} \frac{\sum_{\omega \in \Omega} m_{ia}(c, \omega, \omega^*)}{|\Omega||COMPS|} \quad (8)$$

where for each scenario s $m_{ia}(c, \varpi, \varpi^*)$ and |COMPS| are defined as:

$$m_{ia}(c, \omega, \omega^*) = \begin{cases} 0, \text{ if } (\omega, c) \neq (\omega^*, c) \\ 1, \text{ otherwise} \end{cases} \quad (9)$$

|COMPS| is the set of components tested by the scenario s.

The isolation classification rate is defined based on the construction of a confusion matrix [7, 12]. Fig. 5 shows an example for a "relay" component. In this example, the "relay" can have one healthy (non-fault), and two faulty (stuck_open, stuck_closed) modes. The classification problem then becomes determining what mode the relay will be in.

		Actual State		
	Relay	Non-Faulty	Stuck_open	Stuck_Closed
Predicted State	Non-Faulty	A	F	K
	Stuck_open	B	G	L
	Stuck_Closed	C	H	M

Fig. 5. The confusion matrix [11]

The diagonal values in the confusion matrix capture correctly isolated cases, whereas the off-diagonal elements are incorrect diagnoses. Cumulatively, the confusion matrix summarizes an algorithm's ability to discriminate among multiple fault candidates. In most cases, the confusion matrix is expressed in a normalized form. When normalized, each cell value in the confusion matrix represents an estimate of the probability of that case occurring. In addition, the non-faulty row and column can be removed from the matrix to ensure that the matrix represents a measure of discrimination between faults once it has been determined that a fault is actually present. In this study, a normalized confusion matrix is built for each system component, which summarizes the probabilities of a component's classification cases over a series of scenarios. The cumulative isolation classification rate of a diagnostic algorithm is then calculated by employing a weighted sum over isolation classification rates of individual system components as defined in (8).

7. DIAGNOSTIC ALGORITHMS

The test articles to be evaluated in ADAPT are health management applications from industry, academia, and government. The techniques employed may be data-driven, rule-based, model-based, or a combination of different approaches.

The technologies currently integrated with the testbed include model-based reasoning tools HyDE – Hybrid Diagnostic Engine [13]; FACT – Fault Adaptive Control Technology [14]; TEAMS-RT – Testability Engineering and Maintenance System Real Time [15]; and ADAPT Bayes Net [16]. These tools are from government, academia, and industry, respectively. Each diagnosis algorithm uses the ADAPT API to connect to the ADAPT message server, subscribing to the appropriate commands and data, and publishing the diagnosis results.

Each tool employs different abstraction, modeling, and reasoning methodologies. For example, TEAMS-RT typically discretizes continuous-valued sensor data into pass/fail test results. Cause-effect dependencies in a failure space, multi-signal model that link causes (components) to effects (test results) are used to isolate the fault. In contrast, HyDE uses constraint programming to track the overall system behavior using and a conflict-directed best-first search to efficiently generate fault hypotheses.

Using the benchmarking framework, we aim to explore the advantages and disadvantages of the different approaches to health management to better understand their applicability to different fault types and operational contexts. In the next section, we show how benchmarking results are generated for two of the aforementioned diagnostic algorithms.

8. PERFORMANCE ASSESSMENT

As a case study, we consider two of the aforementioned diagnostic algorithms, denoted algorithm A and algorithm B. Algorithm A has been tested on 16 scenarios, and Algorithm B has been tested on 12 scenarios. These scenarios are summarized in Table 4.

In this paper, we report the results of a *preliminary* study. For illustrative purposes, we have focused on a single type of sensor failure (sensor value stuck at). Experimentation to generate data for the reminder of the fault types is currently underway.

Table 4. The summary of experimental scenarios

Injected scenario			
	Scenerio	Time	Fault
1	459_E140.0	2008-04-10 09:43:11.574 GMT-07:00	E140 Stuck at 0
2	459_E142.0	2008-04-10 09:43:11.574 GMT-07:00	E142 Stuck at 0
3	459_E142.35	2008-04-10 09:43:17.574 GMT-07:00	E142 Stuck at35
4	459_IT167.0	2008-04-10 09:43:11.574 GMT-07:00	IT167 stuck at 0
5	459_LT500.0	2008-04-10 09:43:17.574 GMT-07:00	LT500 stuck at 0
6	459_TE501.800	2008-04-10 09:43:17.574 GMT-07:00	TE501 stuck at 800
7	460_E142.0	2008-04-10 09:46:30.974 GMT-07:00	E142 stuck at 0
8	460_ESH173.0	2008-04-10 09:46:22.974 GMT-07:00	ESH173 stuck at 0
9	460_ESH183.1	2008-04-10 09:46:22.974 GMT-07:00	ESH183 suck at 1
10	460_IT340.15	2008-04-10 09:46:30.974 GMT-07:00	IT340 stuck at 15
11	460_ST515.1300	2008-04-10 09:46:22.974 GMT-07:00	ST515 stuck at 1300
12	460_TE329.800	2008-04-10 09:45:30.974 GMT-07:00	TE329 stuck at 800
13	463_ESH160A.0	2008-04-10 09:58:22.974 GMT-07:00	ESH160A stuck at 0
14	463_ESH283.1	2008-04-10 09:58:30.574 GMT-07:00	ESH183 suck at 1
15	463_ISH336.0	2008-04-10 09:58:30.574 GMT-07:00	ISH336 stuck at 0
16	463_LT500.0	2008-04-10 09:58:30.574 GMT-07:00	LT500 stuck at 0

The results of the experiments with real-world data from ADAPT are summarized in Table 5. Our main observations regarding the results from the experiments are as follows. For algorithm A, in all 16 scenarios, the inserted faults are detected. Algorithm B, on the other hand, correctly detects 7 out of 12 scenarios, and misses the remaining 5 (false negative). Algorithm A also does better in isolation accuracy.

Table 5. The summary of calculated metrics

METRIC RESULTS	ALGORITHM A
Average Fault Detection Time (sec)	8.977
Average Fault Isolation Time (sec)	11.567
Detection False Positive Rate	0.000
Detection False Negative rate	0.000
Fault Detection Accuracy	1.000
Isolation Classification rate	0.972

METRIC RESULTS	ALGORITHM B
Average Fault Detection Time (sec)	N/A
Average Fault Isolation Time (sec)	N/A
Detection False Positive Rate	0.000
Detection False Negative rate	0.417
Fault Detection Accuracy	0.583
Isolation Classification rate	0.954

The temporal metrics (time-to-detect, time-to-isolate) are also calculated and the results are presented in Table 5. In calculating these metrics, we took the average of the algorithm response time over the number of applicable scenarios. Based on the results, on average, Algorithm A takes 8.997 sec to detect a fault, and 11.567 sec to isolate a fault. (For Algorithm B, we couldn't calculate these values due to a processor time clock discrepancy.)

9. CONCLUSIONS

In this paper, we described an architecture and a formal framework developed for benchmarking of diagnostic technologies. The framework defines a number of standardized components, which include a fault catalog, a library of modular test scenarios, and a common protocol for gathering and processing diagnostic data. In addition, it uses 13 benchmarking metrics as a basis of evaluation. These metrics enable the production of comparable performance assessment for different diagnostic technologies. The diagnosed system is the Advanced Diagnostics and Prognostics Testbed (ADAPT), a real-world electrical power system (EPS), developed and maintained at the NASA Ames Research Center.

The process of developing advanced diagnostic applications is important to the verification and validation of hardware systems. ADAPT establishes a problem domain with known fault signatures and the capability to inject failures during operation of the testbed. The electrical power system hardware, together with the software architecture offers many challenges to diagnostic applications such as a multitude of system modes, transient behavior after switching actions, multiple faults, and load-dependent noise.

By implementing and testing different diagnostic approaches on ADAPT, we provide a systematic, realistic, empirical approach to perform V&V of diagnostic software, and aim to improve performance assessment methods and the comparison of diverse health management strategies.

The results presented here are from a preliminary investigation. We have used only a handful of scenarios and one fault type. One of our immediate future goals is to generate more results by expanding the scope of the diagnostic algorithms tested, fault types analyzed, and the variety of experimental scenarios run.

10. ACKNOWLEDGMENT

This work was supported in part by the National Aeronautics and Space Administration (NASA) Aeronautics Research Mission Directorate (ARMD) Aviation Safety Program (AvSP) Integrated Vehicle Management (IVHM) Project. Additionally, this material is based upon the work supported by NASA under awards NCC2-1426 and NNA07BB97C. The authors would like to thank Ann Patterson-Hine and Dougal Maclise (NASA Ames Research Center) for their role in the development of the ADAPT testbed, and David Hall, Adam Sweet, John Ossenfort, Serge Yentus, David Garcia and David Nishikawa, Eric Barszcz, Sriram Narasimhan (NASA Ames Research Center), Alexander Feldman (Delft University), Lukas Kuhn , and Johan De Kleer (Palo Alto Research Center) for their contributions to the benchmarking discussions.

REFERENCES

[1] R. M. Button 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.

[2] S. Poll, 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.

[3] Orsagh R.F., Roemer, M.J., Savage, C.J., and Lebold, M., 2002, "Development of Performance and Effectiveness Metrics for Gas Turbine Diagnostic Techniques", Aerospace 2002 IEEE Conference Proceedings, 2002, Vol6, pp. 2825-2834.

[4] Roemer, M.J., Dzakowic, J., Orsagh R.F., Byington, C.S., and Vachtsevanos, G., 2004, "Validation and Verification of Prognostic Health Management Technologies", Aerospace 2005 IEEE Conference Proceedings, 2005, paper 1344.

[5] Bartys, M., Patton, R., Syfert, m., de las Heras, S., and Quevedo, J., 2006, "Introduction to the DAMADICS Actuator FDI Benchmark Study", Control Engineering Practice, 2006, Vol 14, pp.577-596.

[6] Simon L., J. Bird, C. Davison, A. Volponi, R. E. Iverson, 2008, "Benchmarking Gas Path Diagnostic Methods: A Public Approach", Proceedings of the ASME Turbo Expo 2008: Power for Land, Sea and Air, GT2008, June 9-13, 2008, Berlin, Germany.

[7] SAE (Society of Automotive Engineers) E-32, 2007, "Health and Usage Monitoring Metrics, Monitoring the Monitor", February 14, 2007, SAE ARP 5783-DRAFT.

[8] Depold, H., Siegel, J., and Hull,J., 2004, "Metrics for Evaluating the Accuracy of Diagnostic Fault Detection Systems", ASME GT2004-54144, IGTI Turbo Expo, June 2004, Vienna, Austria.

[9] Depold, H., Rajamani, R., Morrison, W.H., and Pattipati, K.R., 2006, "A Unified Metric for Fault Detection and Isolation in Engines", ASME GT2006-91095, IGTI Turbo Expo, May 2006, Barcelona, Spain.

[10] Metz, C.E., 1978, "Basic Principles of ROC Analysis", Journal of Nuclear medicine, 1978, Vol 8 (4), pp. 283-298.

[11] Kurtoglu, T., O. Mengshoel, S.Poll, 2008, "A Framework for Systematic Benchmarking of Monitoring and Diagnosis Systems", Proceedings of International Conference on Prognostics and Health Management, Oct 6-9, Denver, CO.

[12] U. Lerner, R. Parr, D. Koller, and G. Biswas, "Bayesian fault detection and diagnosis in dynamic systems", In Proc. of the Seventeenth National Conference on Artificial Intelligence (AAAI-00), 2000, pp. 531–537

[13] Narasimhan, S.; Dearden, R.; and Benazera, E. 2004. Combining Particle Filters and Consistency-Based Approaches for Monitoring and Diagnosis of Stochastic Hybrid Systems. 15th International Workshop on Principles of Diagnosis (DX04), Carcassonne, France.

[14] Manders, E. J.; Biswas, G.; Mahadevan, N.; and Karsai, G. 2006. Component-Oriented Modeling of Hybrid Dynamic Systems Using the Generic Modeling Environment. In Proceedings of the 4th Workshop on Model-Based Development of Computer Based Systems.

[15] Deb, S.; Mathur, A; Willitt, P; and Pattipati, K. 1998. Decentralized Real-Time Monitoring and Diagnosis. IEEE Transactions on Systems, Man, and Cybernetics.

[16] Mengshoel J.O, A. Darwiche, K. Cascio, M. Chavira, S. Poll, S. Uckun. "Diagnosing Faults in Electrical Power Systems of Spacecraft and Aircraft". Proceedings of the Twentieth IAAI Conference on Artificial Intelligence (IAAI-08), Chicago, IL, 2008.

BIOGRAPHY

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 computational design tools, design optimization, conceptual design theory, modeling of physical systems, 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 30 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.*