A Review of Diagnostic Techniques for ISHM Applications

Ann Patterson-Hine,
NASA Ames Research Center
Moffett Field, CA

Gordon Aaseng
Honeywell Defense & Space Electronics Systems
Glendale, AZ

Gautam Biswas
Vanderbilt University
Nashville, TN

Sriram Narasimhan
University Affiliated Research Center
Moffett Field, CA

Krishna Pattipati
University of Connecticut
Storrs, CT

Abstract

System diagnosis is an integral part of any Integrated System Health Management application. Diagnostic applications make use of system information from the design phase, such as safety and mission assurance analysis, failure modes and effects analysis, hazards analysis, functional models, fault propagation models, and testability analysis. In modern process control and equipment monitoring systems, topological and analytic models of the nominal system, derived from design documents, are also employed for fault isolation and identification. Depending on the complexity of the monitored signals from the physical system, diagnostic applications may involve straightforward trending and feature extraction techniques to retrieve the parameters of importance from the sensor streams. They also may involve very complex analysis routines, such as signal processing, learning or classification methods to derive the parameters of importance to diagnosis. The process that is used to diagnose anomalous conditions from monitored system signals varies widely across the different approaches to system diagnosis. Rule-based expert systems, case-based reasoning systems, model-based reasoning systems, learning systems, and probabilistic reasoning systems are examples of the many diverse approaches to diagnostic reasoning.

Many engineering disciplines have specific approaches to modeling, monitoring and diagnosing anomalous conditions. Therefore, there is no "one-size-fits-all" approach to building diagnostic and health monitoring capabilities for a system. For instance, the conventional approaches to diagnosing failures in rotorcraft applications are very different from those used in communications systems. Further, online and offline automated diagnostic applications are integrated into an operations framework with flight crews, flight controllers and maintenance teams. While the emphasis of this paper is automation of health management functions, striking the correct balance between automated and human-performed tasks is a vital concern.

1.0 Introduction

System diagnosis is an integral part of any Integrated System Health Management (ISHM) application. During system operation, indications of correct or incorrect functioning of a system may be available. Diagnosis is the process of inferring the cause of any abnormal or unexpected behavior. In complex applications, the symptoms of incorrect (or correct) behavior may be directly observable or they may need to be inferred from other variables that are observable during system operation. Monitoring is a term that is used to denote observing system behavior. The capability for monitoring a system is a key prerequisite to diagnosing problems in the system. Therefore, monitoring requirements will be included in the descriptions of various diagnosis techniques in this paper.

Diagnostic applications make use of system information from the design phase, such as safety and mission assurance analysis, failure modes and effects analysis, hazards analysis, functional models, fault propagation models, and testability analysis. Also of benefit to diagnostic system developers is information on the expected system behavior from operations concepts, which may define multiple operating modes and scenarios for important use cases. In process control and equipment diagnosis applications, topological and analytic models of the nominal system derived from design documents form the core for model-based diagnosis. Depending on the complexity of the monitored signals from the physical system, diagnostic applications may involve straightforward trending and feature extraction techniques to retrieve the parameters of importance from the sensor streams, or they may involve very complex analysis routines, such as signal processing, learning or classification methods. The process that is used to go from monitored system signals to the diagnosis of anomalous conditions varies widely across the different approaches to system diagnosis. Rule-based expert systems, case-based reasoning systems, model-based reasoning systems, learning systems, and probabilistic reasoning systems are examples of the many diverse approaches to diagnostic reasoning.

Diverse disciplines have developed diagnostic approaches using methods relevant to their fields of expertise. For example, once the discipline of systems engineering was established, failure modes and effects analysis and fault tree analysis were commonly used during the design of complex systems such as aircraft and nuclear power plants. Both of these analyses summarize the paths through which failures may propagate in a system, and, consequently, diagnostic dependency models that integrate information from these types of analyses have been employed for a variety of applications. The utility of these methods was recognized by the design community first and was then used by the operations community in the development of on-line monitoring systems. The control systems community developed quantitative model-based methods employing residuals for fault detection and isolation. Residuals are generated by comparing the sensed measurements to the predicted output of a mathematical model that is represented either in the state space or the input-output formulation. The computer science community has developed rule-based methods, which originated from expert systems development, that were initially aimed at medical diagnosis, and qualitative model-based methods that rely on dependency tracking, constraint analysis and qualitative simulations of the dynamics

of system behavior. In model-based methods, abstracted forms of observed behavior are compared to behaviors generated by the qualitative models, and differences are traced by logical inferences and constraint analysis methods to derive the set of potential failure candidates.

Rather than elaborate on the details of specific diagnosis techniques, this paper presents a survey of diagnosis techniques and explains how the different techniques apply in a general framework. In present-day systems, automated diagnostic applications are integrated into an operations framework with system operators, system supervisors and maintenance teams. While the emphasis of this paper is on automation for health management functions, striking a correct balance between automated and human-performed tasks is of vital concern in defining the ISHM architecture. This is especially true for complex, safety-critical systems that can operate in a variety of different modes and in a number of different environments. Special consideration will be given to techniques that have been used in ISHM applications; however, additional techniques will be summarized with references provided for further investigation by the reader.

The challenges for applying diagnostic reasoning technology include determining the best combination of methods for a given system under the constraints of computational resources available, time-to-criticality of the failure behavior, cost of developing the automated system, and the costs of maintaining the automated system over the lifetime of the application.

2.0 General Diagnosis Problem

The complete diagnosis task can be described in three steps that are common to all approaches. Figure 1 illustrates these steps: observation, comparison and diagnosis. The operation of a physical plant is observed using instrumentation appropriate for that application. Sensors commonly found in thermal, electrical, mechanical and fluid systems measure physical characteristics such as temperature, pressure, displacement, strain and vibration. Observations in computer networks include data rates, message retries and physical variables such as voltage, current, and temperature. There is great diversity in sensing instrumentation across system domains. Selecting the best instrumentation suite for complex systems that span multiple physical domains is itself a difficult optimization task. Once the physical sensors have been selected and placed at optimal points in the system,[1] the data acquisition and analysis task can occur during system operation.

The processing of the data and determination of key parameters of interest, extracted from the measured signals, are part of the second step in Figure 1. In this step the observed system state or output is compared to the expected state or output. This step is the fault detection task, i.e., the determination of abnormal behavior. The algorithms used for this comparison range from very simple trending against redline values (known limits) of critical parameters to complex comparisons of measurements to expected

---

[1] This is often determined by factors, such as observability and diagnosability on one hand, and cost and reliability of the measurement on the other.

values predicted using high-fidelity simulations or state estimators. Measured signals often can be noisy and sensors are imperfect; but sophisticated signal processing techniques and statistical testing routines can be employed to keep the false alarm rates to acceptable levels. Neural net diagnosis algorithms use a learned model of the behavior of the system which can be compared against actual behavior observed through sensors.

Many of these comparisons result in knowledge that the system is operating in an abnormal way, and the cause(s) of the off-nominal behavior can be determined by diagnosis algorithms. In several approaches, the detection of off-nominal conditions is tightly integrated with the isolation of the root cause of the problem. In some approaches, the detection of abnormal conditions triggers the fault isolation system. In the following sections, a variety of diagnostic techniques are described and the interactions between these steps are clarified.
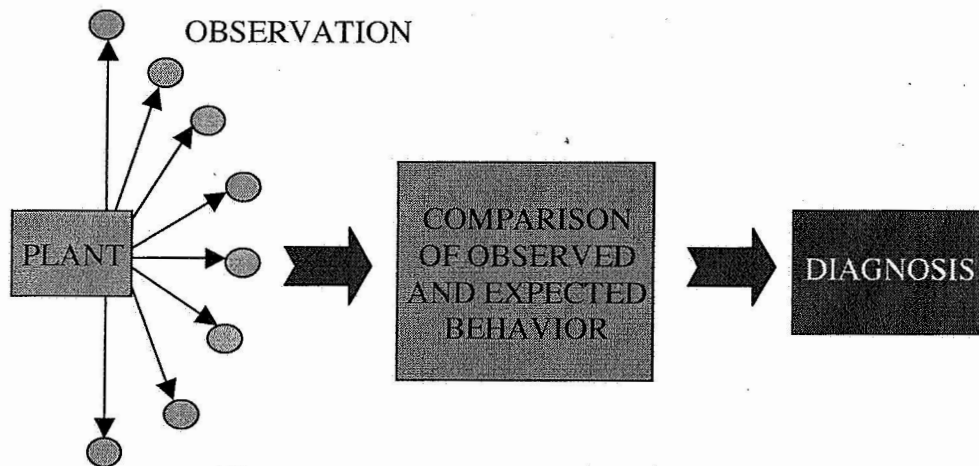
Figure 1. General process for diagnosis.

3.0 Fault Propagation and Impact

A key element of fault diagnosis is the understanding of how faults propagate through a system. In complex engineering applications, systems can be composed of many components and subsystems, and the way these elements interact will affect the way failures propagate within subsystems and across subsystem boundaries. Numerous analyses performed during system design are useful in the early stages of the design of the diagnostic system [1-3, 67]. Failure modes and effects analysis (FMEA) is a bottom-up approach that traces the effects of critical component failures though the system. Fault tree analysis is a top-down approach in which undesirable events are studied to determine all possible causes of that event. In practical applications, it is not feasible to analyze a complex system exhaustively with either technique; therefore, a combination of the top-down and bottom-up analyses is generally advocated. Another useful technique uses directed graphs to analyze component dependencies. The directed graph can be developed from the schematic diagram or functional model, and then failure modes included to the desired level of detail [4]. The nodes represent components or functions, depending on the use of schematics or functional diagrams respectively. The arcs represent the paths of failure propagation through the system.

All failure propagation models can be analyzed to various levels of detail. The desirable level of detail can depend on when in the lifecycle the analysis is performed (models constructed during the early design phases can be at high levels, with more details added as the system designs are firmed up), the available instrumentation (limited visibility into component health can limit the dependency models to higher levels of functionality rather than specific component configurations), and operational requirements (the system needs to be modeled to the line replaceable unit only for applications in which repair or switchover to redundant backup systems is possible). The flexibility to determine the level of modeling detail enhances the usefulness of these techniques. The models can be built with specific purposes in mind, thus saving much time and effort because extensive, detailed analyses are not required. In most cases, if more detail is needed at a later stage, the higher level models can be expanded in the specific areas where more detail is necessary.

Figure 2 (a) shows a segment of a directed graph modeled with a graphical analysis tool called FEAT [5]. This tool provided the capability of coloring the nodes and arcs in the graph to indicate the paths of failure propagation. Red circles indicated that that single event could cause the selected failure, magenta circles indicated that a pair of events represented by the circles needed to happen to result in the selected failure which is shown in green. The results could also be mapped back to a schematic diagram as shown in Figure 2 (b). This was very useful for communicating the results of the analysis to other design team members or project management. In general, diagnostic approaches in this category employ discrete-event diagnosis models and logical dependency tracking methods to isolate faults (root causes), given observed discrepancies (events) [6-7].
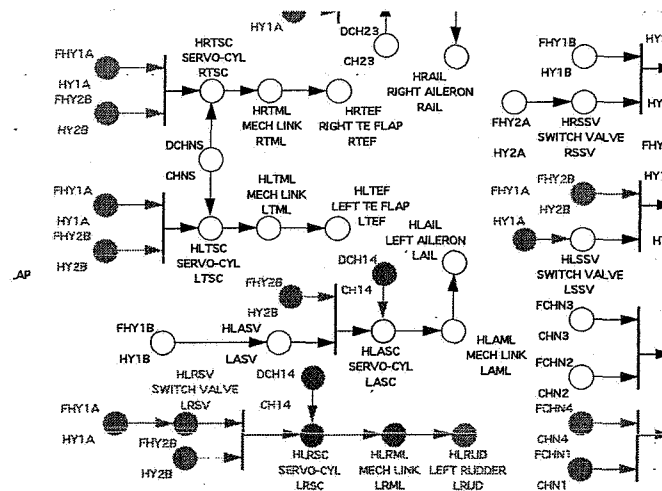

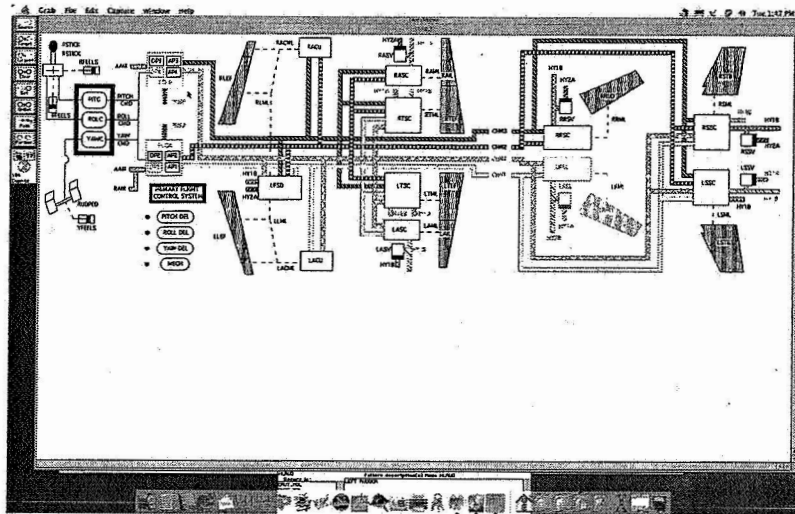**Figure 2(a). Digraph model showing fault propagation.**

Figure 2(b). Fault propagation results shown on schematic diagram.

4.0 Diagnosis Techniques

This section briefly reviews rule-based expert systems, case-based reasoning systems, model-based reasoning systems, learning systems and probabilistic reasoning systems as representative examples of the many diverse approaches to diagnostic reasoning.

4.1 Rule-based Expert Systems

Rule-based expert systems have wide application for diagnostic tasks where expertise and experience are available but deep understanding of the physical properties of the system is either unavailable or too costly to obtain. The procedures that a troubleshooting expert performs can be broken down into multiple steps and encoded into "rules." A rule describes the action(s) that should be taken if a symptom is observed, for instance. A set of rules can be incorporated into a rule-based expert system, which can then be used to generate diagnostic solutions.

Two primary reasoning methods may be employed for generating the diagnosis results. If the starting point is a hypothesis, a backward-chaining algorithm collects or verifies evidence that supports the hypothesis. If the supporting evidence is verified, then the hypothesis is reported as the diagnostic result. In forward chaining, illustrated in Figure 3, the process examines rules to see which ones match the observed evidence. If only one rule matches, the process is simple. However, if more than one rule matches, a conflict set is established and is examined using a pre-defined strategy that assigns priority to the applicable rules. Rules with higher priority are applied first to obtain diagnostic conclusions. A chain of rule firings establishes the diagnostic candidate that is consistent with the observed evidence given the rule set is correct and there are sufficient observations.

The advantages of rule-based systems [8] include an increase in the availability and the reusability of expertise at reduced cost, increased safety if the expertise must be used in

hazardous environments, increased reliability for decision making when the expert system is used as a back-up or tie-breaker in conjunction with human experts, fast response, steady response when a human expert may not be at the peak of performance due to stress or fatigue, and consistent performance across years of operation when human experts may come and go on a project. There is also usually a built-in explanation facility, so that the human operator can understand how the expert system arrived at its conclusion.

A challenging element of this technique is the domain knowledge acquisition step in which the domain expert's understanding of the system and its operation is translated into modular, concise rules, often called the knowledge engineering task [9-12]. There are established procedures and recommendations for soliciting the knowledge of a domain expert or group of experts, and also for managing the large amounts of information that may result from the knowledge acquisition process. The algorithms that attempt to match the current state of the system with rules that pertain to that state are called production systems. Challenges for the production system include resolving conflicts, such as the order in which the rules are matched, and providing supervision over the timing of the rule matching while tracking the current state of the system. Other challenges include determining the completeness, consistency and correctness of the derived rule base for complex systems, and also maintaining the accuracy of a large rule-base over the lifetime of the system. However, for situations in which the diagnosis of failure events in a system is a well-known, stable process and expertise exists, a rule-based expert system may be a good candidate for automating the diagnostic process.

One of the earliest applications of expert systems for diagnosis was MYCIN, developed to diagnose blood infections [13]. MYCIN contained about 450 rules and incorporated a calculus of uncertainty called certainty factors. It was a backward-chaining system. Giarratano and Riley describe the development of the CLIPS (C Language Integrated Production System) originally from NASA Johnson Space Center [8]. Many small systems exist, which are developed for very specific purposes and which contain on the order of several hundred rules. Many troubleshooting tasks fall into this category.
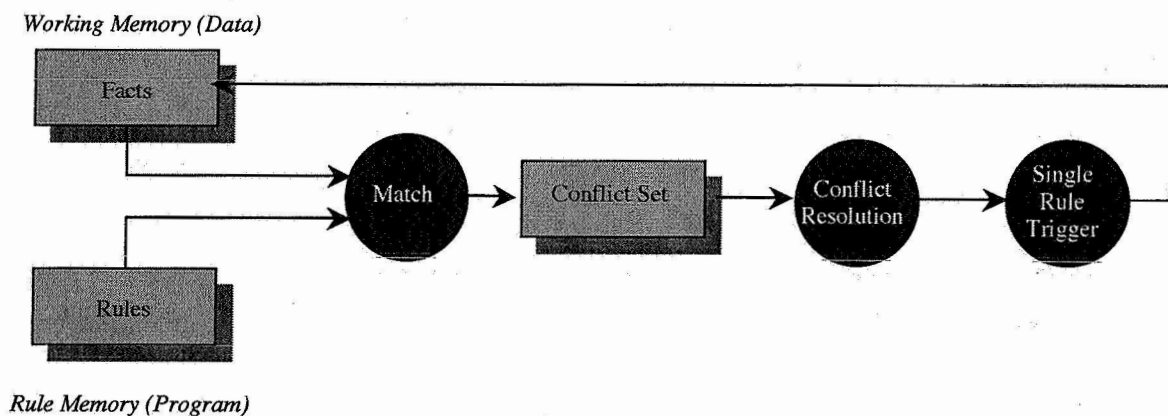
Working Memory (Data)

Rule Memory (Program)

Figure 3. Forward-chaining expert system approach. [14]

4.2 Case-based Reasoning Systems

Case-based Reasoning Systems [15-18] exploit knowledge about solutions developed for past problems to solve current problems. Like rule-based systems, past experience with normal and abnormal behavior of a system are essential to building effective case-based diagnosis systems. In addition, case-based reasoning systems include a learning component which makes possible adaptation of a past solution to fit other, similar situations. This technique is well suited for poorly understood problem areas for which structured data are available to characterize operating scenarios. A case-based reasoning system consists of a case library containing features that describe the problem, outcomes, solutions, methods used and an assessment of their efficacy. A coding mechanism is used to index the case information so that the cases can be organized into meaningful structures, such as clusters, enabling efficient retrieval.

The case-based reasoning architecture entails four basic steps in a cycle shown in Figure 4 [19]:

(1) Retrieval – given a new, indexed problem, retrieve the best past cases from memory.

(2) Reuse – find the difference between the past and current case and transfer or modify the old solution to conform to the new situation, resulting in a proposed solution.

(3) Revise – determine whether the proposed solution is successful and give a confirmed solution. If the solution fails, explain the failure, learn how to avoid repeating it, and repair the solution; if the solution succeeds, go to step 4.

(4) Retain – incorporate the new solution into the existing knowledge.

An extensive use of case-based reasoning is in remote diagnosis on locomotives to quickly identify failures that have occurred or are about to occur and that may result in a locomotive stranded on the tracks due to equipment failure. A vast amount of historical fault logs and repair history of locomotives is available. A condition-based reasoning system was developed for this area, and has been in continuous use since 1995 [20]. Gas turbine diagnostics are performed at General Electric using this technique as well. When a turbine trips, the condition-based reasoning system is used to automate the data review, hypothesis generation and hypothesis confirmation tasks in the trouble-shooting process, and assist the user when it does not have confidence in a single cause [21]. Other applications are discussed in [22-24]. Case-based systems may work well when the diagnosis task is performed in conjunction with a human operator. When unusual situations occur, the system may make suggestions, but the operator uses these as a guide and runs additional tests to verify the correctness of the proposed diagnostic hypothesis.
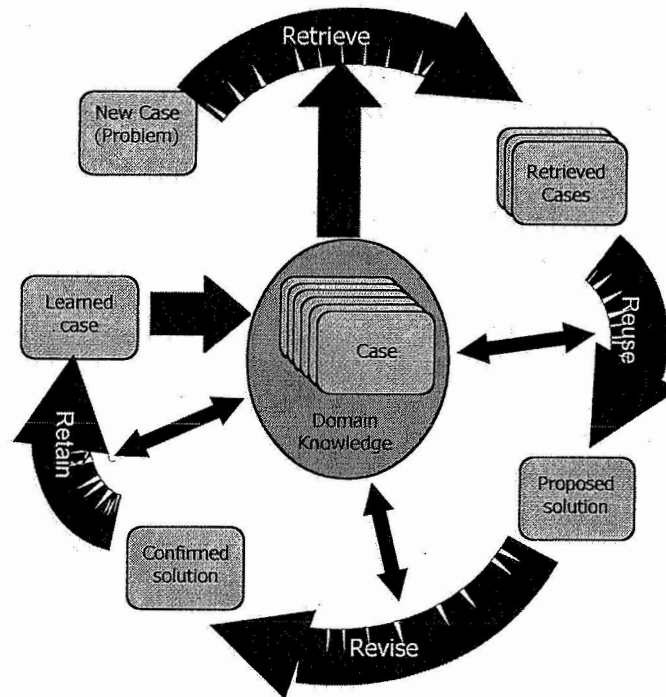
Figure 4. The Retrieve-Reuse-Revise-Retain process [19].

4.3 Learning Systems

Learning systems are data-driven approaches that are derived directly from routinely-monitored system operating data (e.g., calibration, power, vibration, temperature, pressure, oil debris, currents or voltages). They rely on the assumption that the statistical characteristics of the data are stable unless a malfunctioning event occurs in the system. That is, the *common cause variations* are entirely due to uncertainties and random noise, whereas *special cause variations* (e.g., due to faults) account for data variations not attributed to common cause. The strength of data-driven techniques is their ability to transform the high-dimensional *noisy* data into lower-dimensional *information* for detection and diagnostic decisions. The data-driven methods provide the ability to handle highly collinear data of high dimensionality, substantially reduce the dimensionality of the monitoring problem, and compress the data for archiving purposes. In addition to providing monitoring methods of their own, data-driven approaches facilitate model building via identification of dynamic relationships among data elements. The main drawback of data-driven approaches is that their efficacy is highly dependent on the quantity and quality of system operational data.

The engineering processes needed to relate system malfunctioning events using a data-driven diagnosis approach typically involve the following steps.

1. **Determine the High-Impact Malfunctions:** From historical data, understand the nature of real and potential faults, their location, their characteristic symptoms, and their severity (measured in terms of safety, mission criticality and cost).
2. **Data Selection, Transformation, De-noising and Preparation:** Data cleaning and preprocessing (e.g., data normalization and de-noising) and data reduction and representation (e.g., finding dominant directions, clustering of data, recognizing events independent of scale) constitute 50-75% of the effort in building data-driven diagnosis models. When the data set is noisy and includes more variables than necessary, methods for selecting data records for initial data exploration and model building (based, for example, on empirical statistics and correlations) are important. Data transformation techniques include component scaling, histogram equalization and sample-by-sample nonlinearities. De-noising is typically performed by lowpass, highpass, bandpass, and bandstop filters, both windowed finite impulse response (FIR) and any of Butterworth, Chebychev, or elliptic infinite impulse response (IIR) filters. These may be run efficiently on a block of data subsequent to their design. The data selection, normalization and filtering steps culminate in a data preparation phase that covers all activities to construct the final data sets for classification and model building.
3. **Data Processing Techniques:** The data-driven classification approaches are numerous and are selected based on competitive evaluation and possibly cooperative fusion. These procedures have to be carefully tuned to minimize false alarms while improving their classification capability. The procedures should have the capability to detect trends and degradation and assess the severity of a failure for early warning. Among the myriad of learning-based techniques, principal component analysis (PCA), partial least squares (PLS) and support vector machines (SVM) provide consistently accurate diagnosis across a range of applications, including chillers, automotive and text categorization tasks [32-37].
4. **Testing and Validation:** Testing and validation of models is perhaps the most important step in ensuring the quality and robustness of the models on live data. These methods test models using leave-one-out, N-fold cross validation (train on (N-1) sets and test on one set in a round-robin fashion) or bootstrap techniques. This process is repeated to adapt the models as the data accumulate over time.
5. **Fusion:** A diagnostic system has the potential for higher diagnostic accuracy if it is capable of fusing results from multiple diverse classifiers to estimate fault severity and to evaluate the health of the integrated system.

The data processing techniques for diagnosis can be broadly divided into four major categories:

Multivariate Statistical methods

Classical least squares regression techniques are inappropriate for handling noisy and highly correlated data, since the least squares problem will invariably be ill-conditioned, resulting in poor predictions. The techniques of principal components analysis (PCA), and partial least squares (PLS) surmount these problems by projecting the multivariate data onto a space of as few as two or three dimensions.

PCA is a multivariate statistical modeling technique that finds the directions of significant variability in the data matrix by forming combinations of existing variables to orthogonal principal components (PCs). The data matrix is created with replicated samples of data (batches) as rows and monitored variables as columns. When the data contain dynamic information, the current variables will depend on the past values. Therefore, in a multi-way PCA (MPCA), the data are arranged in a three-dimensional array (a tensor) of batches by variables by time. Then, the data are centered and scaled, a multi-way PCA is performed on the tensor, and the first r scaled right singular vectors (that explain 90% or more of the variability in the data) are selected as the loading vectors. When new data are received, the r score vectors (principal components) in a lower-dimensional space are formed by computing the inner product of the data with each of the loading vectors. Hotelling's T^2 (sum of squares of the scores), which measures the variations in the score space, has a c^2 distribution. The T^2 statistic can be interpreted as measuring the normal variations of system operation and the violation of a threshold on T^2 would indicate that the system has malfunctioned. Similarly, the sum of squares of residuals Q measures the random variations of the nominal system behavior. A violation of the threshold on the Q statistic would indicate that the random noise has significantly changed. These two statistics, along with their respective thresholds, yield a cylindrical in-control region for normal system operation.

PLS (also known as projection to latent squares) and multi-way PLS are similar to the projection techniques of PCA and MPCA. PLS reduces the dimensionality of the input and output spaces to find the latent vectors for the input and output spaces which are most highly correlated, i.e., those that not only explain the variation in the input, but the variation which is most predictive of the output. In the context of diagnosis, PLS builds regression models between the monitored variables and the fault classes.

Signal Analysis Methods

Many measured signals exhibit oscillations that have either harmonic or stochastic nature or both. Signal analysis methods include a wide menu of spectral and statistical manipulation primitives such as filters, harmonic analyzers, auto and cross-correlation functions, fast Fourier transform (FFT), multi-resolution decomposition ("wavelets"), root mean square (RMS) values, time synchronous average residue (TSAR) and kurtosis. These methods are used in the data preparation phase or as data processing modules when coupled with statistical hypothesis testing methods (e.g., cumulative sum, generalized likelihood ratio test (GLR).

Machine Learning

Machine learning techniques include nonlinear regression, support vector machines (SVM), probabilistic neural networks, decision trees, single and multi-layer perceptrons, radial basis functions, k-means clustering, learning vector quantization, Bayesian networks, hidden Markov models, instance-based classifiers, self-organizing feature maps and fuzzy logic. We will briefly describe only SVM because it has been found to perform consistently well across a range of applications.

Support vector machines (SVM), as a supervised statistical learning theory, has gained popularity in recent years for classification and regression because of its four distinct advantages. First, SVM is a universal learner with proper selection of the kernel function. Second, it has the ability to learn with a small amount of training data, even when the number of features (terms) is large. Third, SVM is well suited for sparse computations. Finally, most categorization problems are linearly separable in a higher-dimensional space. The SVM has been successfully employed in a variety of applications, such as pattern recognition, multiple regression, nonlinear model fitting and text categorization.

The essential idea of SVM classification is to transform the input data to a high-dimensional feature space and find an optimal hyperplane that maximizes the margin between the classes. The group of examples that lie closest to the separating hyperplane is referred to as support vectors. For SVM regression, the input is first mapped onto high-dimensional feature space using nonlinear mapping (the kernel function), and then a linear regression is performed in this feature space.

The block diagram for designing a representative fault detection and isolation (FDI) scheme using a learning system approach is shown in Figure 5. We arranged the FDI scheme as a three step process: fault detection, fault isolation using statistical and machine learning techniques and fault severity estimation using MPLS.

Chaos Engineering

Recently, chaos engineering has found a number of applications in home appliances (e.g. oil fan heaters, air-conditioners, dish washing dryers and washing machines) and in tap water quality prediction. The key idea in the context of fault diagnosis is that there is a distinct trajectory of features associated with a fault [31] and that it can be inferred from sensed observations.

4.4 Model-based Reasoning

Model-based reasoning is a broad category that describes the use of a wide variety of engineering models as the foundation for the knowledge and the techniques applied for diagnosis. In parallel developments, with the advent of powerful embedded processors, 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 dynamic systems [43]. Researchers in the computer science community for Model-Based Diagnosis (MBD) employed a model of the system configuration and behavior of the system for the diagnosis task [38-39]. In process control communities, state equations and transfer function representations serve as the system model [40,44]. Practical systems engineering approaches have employed fault-propagation graphs as the system model for diagnostic reasoning [41,45,46]. In all of these cases, the sensed state of the system is
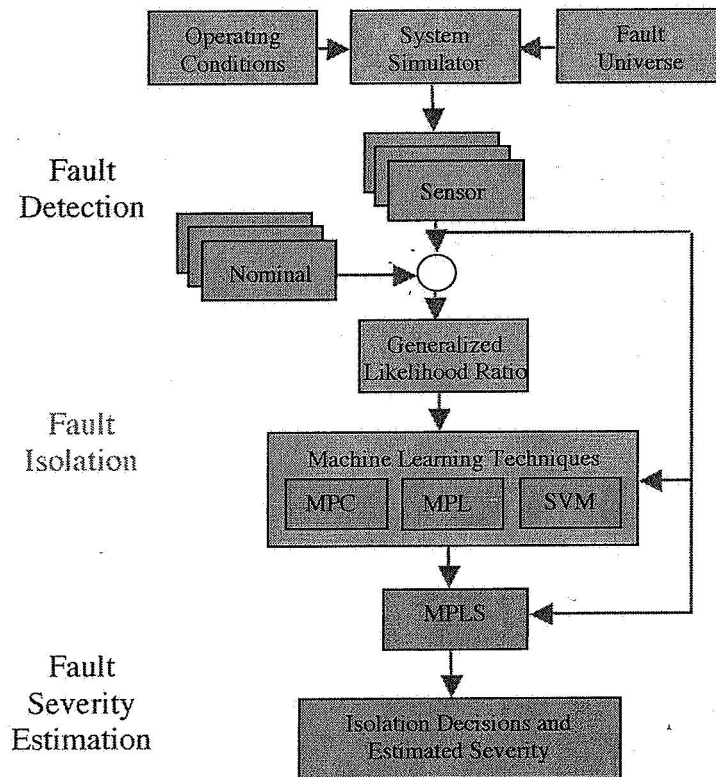
Figure 5. Block diagram of a Data-driven FDI scheme

compared to what is expected (the monitoring and fault detection task) and a discrepancy implies the occurrence of an anomalous condition.

In the computer science or Artificial Intelligence approaches to diagnosis, the diagnosis algorithm reasons about the differences between predictions (made by a functional model of the system) and observations (obtained from the actual system). Figure 6 illustrates the approach. Comparing the predicted and actual behavior may result in discrepancies that imply the occurrence of faults (malfunctioning events). The detected discrepancies are analyzed in the context of the system model to generate fault hypotheses and refine them as more information becomes available [47], as shown in the diagram. Discrepancies are analyzed in one of two ways: (i) discrepancies are interpreted as a violation of the constraints that define system behavior, and relaxation of the constraints implicates faulty components [38,39,48], and (ii) logical analysis of Boolean constraints and analysis of the inconsistencies in the constraints produce fault hypotheses. These approaches, developed by the AI Diagnosis (DX) community, are termed consistency-based approaches to diagnosis. Most work on qualitative fault diagnosis applies to static systems (e.g., combinational circuits) or systems in steady state. There is some work on qualitative fault diagnosis of dynamic systems based on analysis of fault signatures [49,50,51].
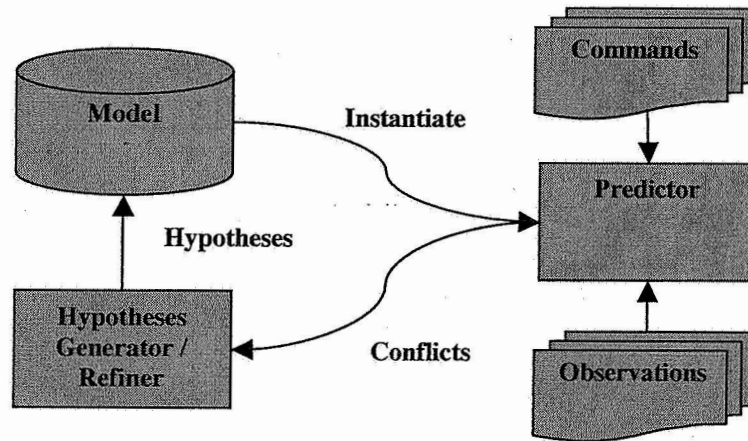
Figure 6. Consistency-based approach in model-based reasoning.

Fault detection may also result from comparing the system measurements with models of the system that describe its behavior under abnormal conditions. Detections in this case are the result of matching observations to predicted behavior in the presence of faults (fault observers). Limit checks are a simple example of this type of detection. Once the presence of a fault is identified, simple reasoning algorithms isolate the fault to the root cause. The fault observers are constructed such that simple logical analysis of the outcomes of a set of observers uniquely identifies the root cause or the diagnostic hypothesis. In the systems engineering approach, models of expected fault propagation paths, also called causal models, are used to determine the cause of anomalous behavior [7]. The interrogation of the fault propagation graph is very efficient. This representation also enables explanation of the reasoning process that is close to human reasoning [52]. In this technique, the complexity of nominal and abnormal system behavior is represented in the monitoring (detection) algorithms.

The process control community has developed approaches based on dynamic quantitative models typically represented as a set of differential equations or a set of input-output transfer functions. These are typically nominal models of system behavior, and when measured behavior is analyzed with various filters, precise numerical vectors called residuals are produced (see Figure 7). Residuals are numerical fault indicators in this process. Early work on residual generation and analysis methods included the use of a bank of Kalman filters (called "matched filters"). The innovation (i.e., the prediction error) of the Kalman filter was used as a fault detection residual (mean = 0, if no fault; mean $\neq$ 0 if there is a fault). A bank of filters (one for each potential fault candidate) was used for fault isolation [53,54]. Further advances in observer-based fault analysis included the design of "unknown input" observers, where the fault residuals were decoupled from inaccuracies in the model and a limited number of input disturbances to the system [55,56,57,58]. This decoupling made fault isolation techniques more robust, sensitive and precise.

In general, most of the observer-based techniques apply well to linear dynamic systems, but they do not extend as easily to non-linear systems with complex behaviors. Recently,

there has been work on the design and implementation of nonlinear observers (e.g., Garcia and Frank [59]). A number of approaches have adopted hybrid methods for diagnosis (see for example, the techniques discussed in the last section). They combine analytic, neural, fuzzy, statistical, and spectral methods for fault detection and isolation. Other innovative approaches involve combining statistical fault detection and symbol generation with qualitative fault signature methods and quantitative parameter estimation methods to obtain precise diagnostic results, while avoiding the computational complexity of most analytic methods [50,60].
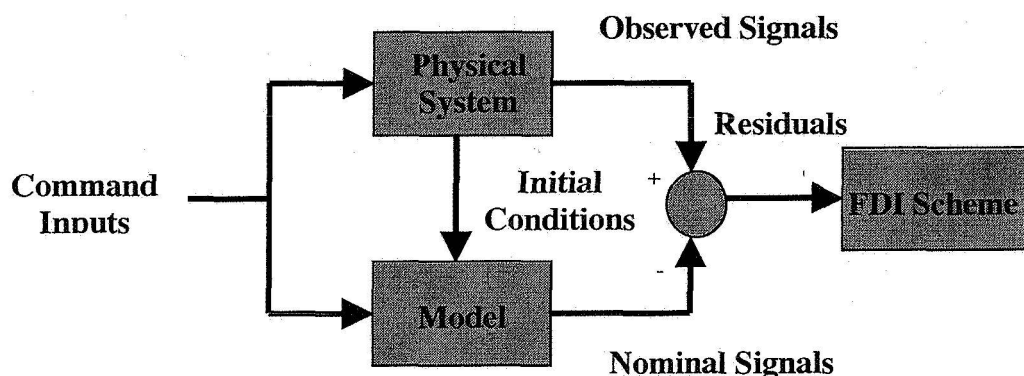

Figure 7. Generation of residuals.

Recently, there have been efforts to compare and combine the consistency-based approaches developed by the DX community with approaches based on engineering disciplines, such as control theory and statistical decision making, used by the Fault Detection, Isolation and Recovery (FDIR) community. In recent years, there have been joint conferences and workshops as well as publications that aim to bridge the gap between the languages and approaches used by these two communities. Interestingly, these activities have been called the BRIDGE community. An excellent source of information about recent work in this area is the special issue of IEEE Systems, Man, and Cybernetics, Part B in October 2004 [61].

Model-based reasoning applications include diagnostics and troubleshooting in the electrical power industry [62], spacecraft such as Deep Space 1 and Earth Observing 1 [63-64], and International Space Station [65, 66].

5.0 Automation considerations for diagnostic systems

Determining the diagnostic and control functions to be automated requires understanding of the effects that the automation will have on operations occurring years in the future. Functions must either make the system safer by performing functions faster, more reliably or more accurately than crews can, or they must maintain a safety level at a significantly lower cost, in order to warrant inclusion in the design. Determining the cost of future operations, with varying levels of automation, is a complex but necessary task for achieving affordable, reliable, safe and effective ISHM programs. Safety and cost

models will provide the basis for deciding if functions are automated or manual, on-board or off-board, or real-time or off-line.

Selecting the right method for health state determination and automation is a complex decision. First, the design organization needs a process for deciding what to automate. Then, the method for performing the automation can be selected. The decision should be to automate a diagnostic function if:
- the automated system can provide valuable information that could not be obtained at all, or quickly enough to be useful, without the automated system;
- the automated system offers significant improvements in the quality of information over human-performed diagnostic activities, such as increased accuracy or consistency; or
- the automated system can perform the diagnostic function at a lower cost than human-performed diagnosis.

Much of the activity of flight crews and supporting teams involve managing the health of the vehicle by monitoring data, watching for off-nominal indications, diagnosing the cause of abnormalities, and mitigating the effects of failures. Maintenance and launch preparation organizations spend much of their time either looking for indications of failure or proving that no off-nominal conditions are present. Any automated system should be designed and built with full understanding of the benefits to the program to be provided by the automated diagnostic system.

Diagnostics designed to improve safety and mission assurance should be able to demonstrate their degree of improvements. Analysis of the benefits of the diagnostic system must be integrated with the hazard analysis, Probabilistic Risk Assessment (PRA) and other safety metrics to show a quantifiable improvement in the assessments. Crew monitoring and procedural training is inadequate for failures that occur with little or no warning and result in catastrophic consequences, such as a high-speed turbopump disintegration resulting in a launch vehicle explosion. Automated failure detection and initiation of crew escape systems for some failures is necessary for crew safety for certain types of failures. Figure 8 shows a conceptual matrix for determining if detection and response must be automated or if manual or collaborative responses are sufficient for assuring the safety of the crew.

Space vehicle maintenance and launch preparation operations are complex, lengthy and expensive. Much of the activity involves testing to assure that the vehicle and support systems are in fully nominal conditions and ready to launch. These activities involve both detecting failures on the ground and verifying that there are no failures or incipient conditions that could pose flight hazards. Automation of these detection and verification activities holds significant promise for reducing costs and shortening the launch flow timelines, as well as improving the quality of the results of diagnostic and verification procedures and testing. However, it is not always clear how automation will affect the overall cost of operations, either by reducing the size of the workforce, shortening the launch flow timeline, increasing the flight rate possible within the system or improving the mission assurance probabilities. These questions involve very complex analysis of
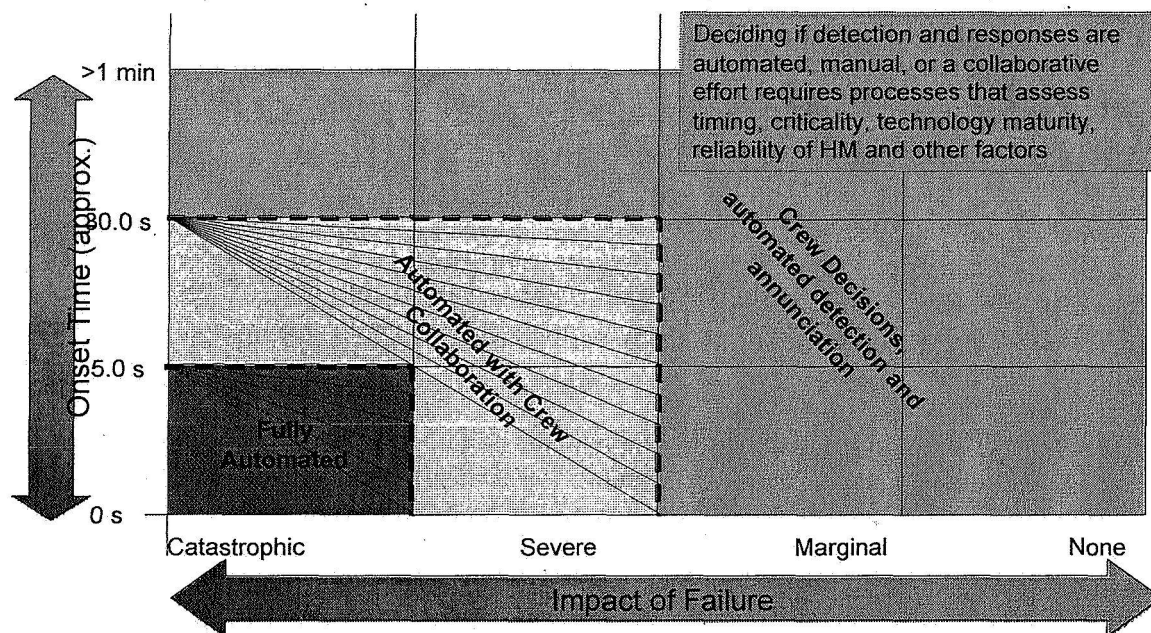
Figure 8 Conceptual matrix for automation decision-making.

operations, and determining the impact that a particular diagnostic or automation application will have on program cost and reliability figures of merit have proven elusive. Operations and cost analysis, preferably using program-level modeling and simulation to determine the high-value targets for automation in the launch flow, is necessary for making correct decisions on which health management functions to automate. A major consideration is the cost of building and maintaining the automated system, compared to the cost of training and supporting the human teams and providing them with the tools needed to perform the diagnostic functions.

Similar analyses are warranted for flight operations to determine how automated diagnostics can affect the cost of setting up and supporting the flight control team and training the flight crew. The Mission Control Center (MCC), Mission Evaluation Room (MER), contractor facilities and the organizations that maintain the facilities, build the tools and support operations constitute a large workforce. Flight crew training involves extensive drilling in recognizing emergency conditions and executing emergency procedures, often exacerbated by the complex and sometimes confusing information presented to the crew. Determining how much automated diagnostics, automated decision support tools, procedure management applications and related systems can impact these operations organizations is a necessary element of the systems engineering related to health management.

The technologies available to the spacecraft that will implement the Vision for Space Exploration are far advanced from what previous programs had to work with, as described throughout the preceding sections. These applications can be costly to design, implement and test, and are themselves subject to failure. Careful systems engineering

must accompany the use of these technologies to assure that their deployment improves crew safety, mission assurance or cost reductions.

6.0 Summary and Conclusions

Automated diagnostic applications have been implemented with a wide variety of techniques and in many diverse domains, as surveyed in this paper. As automation becomes more widespread, the importance of verification and validation of both hardware and software components becomes increasingly important. This paper has focused on the algorithms. Verification methods are discussed in a companion paper in this Forum.

The reliability of diagnoses is highly dependent on the accuracy of the sensed measurements and sensor and instrumentation issues are addressed in another companion paper. As applications become more complex to meet the requirements of increasing autonomy, system level information fusion techniques will need to fuse diagnostic information from a variety of sources. Information fusion techniques are addressed in a third companion paper. Diagnostic algorithms are at the heart of every health management application, and selecting the most appropriate techniques to perform diagnostic reasoning can be quite challenging. The challenge for the future is developing generic diagnostic architectures that can use a variety of techniques and which can scale to cover critical events for an entire system.

7.0 Acknowledgements

The authors appreciate the many helpful comments from the Forum reviewers and from Lee Brownston and Peter Robinson, members of the RIVA group at NASA Ames. We would like to thank Prof. George Vachtsevanos for his list of references and example applications used in the case-based reasoning section.

8.0 References

1. Henley, Ernest J. and Kumamoto, Hiromitsu, <u>Designing for Reliability and Safety Control</u>, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1985.
2. Vesely, W.E., et al, "Fault Tree Handbook, NUREG-0492, http://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0492/, 1981.
3. Tumer, I.Y. and R.B. Stone, *Mapping Function to Failure During High-Risk Component Development.* Research in Engineering Design, 2003. **14**: p. 25-33.
4. Sacks, Ivan J., "Digraph Matrix Analysis," *IEEE Transactions on Reliability*, vol. R-34, no. 5, pp 437-446, December, 1985.
5. Stevenson, Robert W., Miller, James G., Austin, Michael E., "Failure Environment Analysis Tool (FEAT) Development Status," *AIAA Computing in Aerospace VIII Conference*, AIAA 91-3803, Baltimore, MD, 1991.
6. Kirby, S., et al, "Real-time Automated Failure Analysis for On-orbit Operations." *Proceedings of Applications of Artificial Intelligence 1993,* SPIE Proceedings Volume 1963, Orlando, FL.

7. Deb, Somnath, Pattipati, Krishna, and Shrestha, R., "QSI's Integrated Diagnostics Toolset," *Proc. IEEE Autotestcon 1997*, Anaheim, CA, pp. 408-421.
8. Giarratano, Joseph C. and Riley, Gary D., Expert Systems: Principles and Programming, Fourth Edition, PWS Publishing Company, Boston MA, 2004.
9. Jackson, Peter, Introduction to Expert Systems, Third Edition, Addison Wesley, 1998.
10. Brachman, Ronald and Levesque, Hector, Knowledge Representation and Reasoning (The Moran Kaufmann Series in Artificial Intelligence), Morgan Kaufmann, 2004.
11. Patterson, Dan W., Introduction to Artificial Intelligence and Expert Systems, Prentice Hall, 1990.
12. Edmunds, Robert A., The Prentice Hall Guide to Expert Systems, Prentice Hall Trade, 1988.
13. Buchanan, B.G. and Shortliffe, E.H., editors, Rule-based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, Addison-Wesley, 1984.
14. Jones, M. Tim, AI Application Programming, Second Edition, Charles River Media, Inc., 2005.
15. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, *AI- Communications*, 7 (i), pp. 39-59. 1993.
16. Winston, P. H.: Artificial Intelligence, 3rd ed., Addison-Wesley Publishing Co., 1993
17. Stefik, M.: Introduction to Knowledge Systems, Morgan Kaufmann Publishers San Francisco, CA, 1995.
18. Kolodner, J. L.: Case-Based Reasoning, San Mateo, CA, Morgan Kaufmann Publishers, Inc 1993.
19. Berenji, Hamid, Wang, Yan, Jamshidi, Mo, Vachtsevanos, George, and Vengerov, David, "Gated Experts Neural Networks for Prognostics," Technical report IIS-05-01, May 20, 2005.
20. Varma, A. and Roddy, N., "ICARUS: A Case-Based System for Locomotive Diagnostics," *Engineering Applications of Artificial Intelligence Journal*, 1999.
21. Devaney, Mark and Cheetham, Bill, "Case-Based Reasoning for Gas Turbine Diagnostics," *AAAI 2005*.
22. Lehane, M., Dube, F., Halasz, M., Orchard, R., Wylie, R., Zaluski, M., Integrated Diagnostic System (IDS) for Aircraft Fleet Maintenance, *Proceedings of the AAAI '98 Workshop: Case-based Reasoning Integrations*, Technical Report WS-98-15. Madison, Wisconsin, USA. July 27, 1998. pp. 91-95. NRC 43577.
23. Saxena, A., Wu, B., Vachtsevanos, G.: Integrated Diagnosis and Prognosis Architecture for Fleet Vehicles Using Dynamic Case Based Reasoning, appearing in *IEEE Autotestcon* 2005.
24. Cookson R. L.:An evaluation of case-based reasoning for fault diagnosis, PhD Dissertation: The University of New Brunswick (1997): http://digitalcommons.hil.unb.ca/dissertations/AAIMQ23785/.
25. Duda, R.O., Hart, P.E., and Stork, D., *Pattern classification*, John Wiley & Sons, New York, 2000.

26. Bishop, C.M., *Neural Networks for Pattern Recognition*, Clarendon Press, Oxford, 1997.

27. Cherkassky, V., and Mulier, F., *Learning from data, concepts, theory and methods*, John Wiley & Sons, New York, 1998.

28. Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J., *Classification and Regression Trees*, Wadsworth, California, 1984.

29. Quinlan, J.R., *C4.5: Programs for Machine Learning*, San Mateo, CA: Morgan Kaufmann, 1993.

30. Jordan, M.I., (Ed.)., *Learning in Graphical Models*, The MIT Press, 1999.

31. Iokibe, T., " Industrial Applications of Chaos Engineering," http://www.riccx.com/e/paper/1997.8.pdf

32. Namburu, S.M., H. Tu, J. Luo and K.R. Pattipati, "Experiments on Supervised Learning Algorithms for Text Categorization," *IEEE Aerospace Conference*, Big Sky, Montana, March 2005.

33. Choi, K., Namburu, S.M., Azam, M.S., Luo, J., Pattipati, K.R., and Patterson-Hine, A., "Fault Diagnosis in HVAC Chillers," *IEEE Instrumentation & Measurement Magazine*, Vol. 8, No. 3, pp. 24-32, August 2005.

34. Luo, J., Tu, F.,. Azam, M., Pattipati, K.R., Willett, P., Qiao, L., and Kawamoto, M., "Intelligent Model-based Diagnostics for Vehicle Health Management," *SPIE Aerosense*, Vol. 5107, *Track: Signal and Image Processing, System Diagnosis and Prognosis: Security and Condition Monitoring Issues III*, Orlando, FL, April 2003.

35. J. Luo, K.R. Pattipati, L. Qiao, and S. Chigusa, "Agent-based Real-time Fault Diagnosis," *2005 IEEE Aerospace Conference*, Big Sky, Montana, March 2005.

36. Bronson, R.J., Depold, H., Rajamani,R., Deb, S., Morrison, M., and Pattipati, K.R., "Optimal Data Normalization for Engine Health Monitoring," *Proceedings of GT 2005: ASME Turbo Expo 2005*, Reno-Tahoe, Nevada, June 6-9 2005.

37. Morrison, William, Pattipati, Krishna, Morrison, John, Hoffman, Richard, and Slade, James, "Intelligent Self-Evolving Prognostic Fusion," Interim Progress Report, NASA Contract NNA05AC24C, 2005.

38. Hamscher, W., Console, L., and De Kleer, J,, Readings in model-based diagnosis, Morgan Kaufmann Publishers, San Mateo, CA, 1992.

39. Reiter, R., "A theory of diagnosis from First Principles, *Artificial Intelligence*, vol. 32, no. 1, pp. 57-96, 1987.

40. Korbicz, Jozef, Koscielny, Jan M., Kowalczuk, Zdzislaw, and Cholewa, Wojciech, Fault Diagnosis: Models, Artificial Intelligence, Applications, Springer 2004

41. Deb, S., Pattipati, K., Raghavan, V., Shakeri, M., and Shrestha, R., "Multi-signal 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.

42. Gentil Sylviane, Montmain, Jacky, and Combastel, Christophe, "Combining FDI and AI Approaches Within Causal-Model-Based Diagnosis," *IEEE Trans. On Systems, Man, and Cybernectics–Part B: Cybernetics*, vol. 34, no. 5, p. 2207-2221, October 2004

43. Patton, R., P. Frank, and R. Clark, <u>Fault Diagnosis in Dynamic Systems: Theory and Applications</u>, Prentice Hall, Inc., Hertfordshire, UK, 1989.

44. Himmelblau, D.M., <u>Fault Detection and Diagnosis in Chemical and Petrochemical processes</u>, Elsevier, Amsterdam, The Netherlands, 1978.

45. Kramer, M.A. and B. L. Palowitch Jr, "A rule-based approach to fault diagnosis using the signed directed graph," *AIChE Journal*, vol. 33, no. 7 , pp. 1067 – 1078,

46. Padalkar S., Sztipanovits J., Karsai G., Miyasaka N., Okuda K.: Real-Time Fault Diagnostics, *IEEE Expert* , 6, 3, pp. 75-85, 1991.

47. Williams, B.C. and P.P. Nayak, "A Model-based Approach to Reactive Self-Configuring Systems," Workshop on Logic-Based Artificial Intelligence, Washington, DC, June 14-16, 1999.

48. deKleer, J. and B.C. Williams, "Diagnosing multiple faults," *Artificial Intelligence*, vol. 32, pp. 97--130, 1987.

49. Rose, P. and M.A. Kramer, "Qualitative Analysis of Causal Feedback," *Proc. Ninth Nat'l Conf. Artificial Intelligence*, MIT Press, Cambridge, Mass., pp. 817-823, 1991.

50. Mosterman, P.J. and G. Biswas, "Diagnosis of Continuous Valued Systems in Transient Operating Regions," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 29, no. 6, pp. 554-565, Nov. 1999.

51. Trave-Massuyes and R. Milne, "Gas-Turbine Condition Monitoring Using Qualitative Model-Based Diagnosis," *IEEE Expert: Intelligent Systems and Their Applications*, vol. 12 , no. 3, pp. 22-31,   May 1997.

52. Gentil, S., J. Montmain, and C. Combastel, "Combining FDI and AI Approaches withinCausal-Model-based Diagnosis," *IEEE Transactions on Systems, Man abnd Cybernetics, Part B*, vol. 34, no. 5, pp. 2207-2221, Oct. 2004.

53. Mangoubi, R.S., <u>Robust Estimation and Failure Detection</u>, Springer Verlag, London, 1998.

54. Gustafsson, F., <u>Adaptive Filtering and Change Detection</u>, John Wiley and Sons, 2001.

55. Frank, P.M., "Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy — a survey and some new results," *Automatica (Journal of IFAC)*, vol.26 no.3, pp.459-474, May. 1990

56. R.J. Patton and J. Chen, "Observer-based Fault Detection and Isolation: Robustness and Applications," *Control Engineering Practice*, vol. 5, pp. 671-682, 1997.

57. Gertler, J., "Fault Detection and Isolation using Parity Relations," *Control Engineering Practice*, vol. 5, pp. 653-661, 1997.

58. Isermann, R. and P. Balle,"Trends in the Application of Model-based Fault Detection and Diagnosis of Technical Processes," *Control Engineering Practice*, vol. 5, pp. 709-719, 1997.

59. Garcia, E.A. and P.M. Frank, "Deterministic Nonlinear Observer-based Approaches to Fault Diagnosis: A Survey," *Control Engineering Practice*, vol. 5, pp. 663-670, 1997.

60. Patton, R J; Chen, J; Nielsen, S B., "Model-based methods for fault diagnosis: some guide-lines," *Transactions of the Institute of Measurement and Control*, vol. 17, no. 2, pp. 73-83. 1995

61. Biswas, G., M.O. Cordier, J. Lunze, L. Trave-Massuyes, and M. Staroswiecki, "Diagnosis of Complex Systems: Bridging the Gap between the FDI and DX communities," Guest Editorial, special issue of *IEEE Trans. on Systems, Man, and Cybernetics, Part B,* vol. 34, no. 5, pp. 2139-2142, Oct. 2004.

62. Azam, Mohammad, Tu, Fang, Pattipati, Krishna, and Karanam, Rajaiah, "A Dependency Model Based Approach for Identifying and Evaluating Power Quality Problems, " *IEEE Trans. On Power Delivery,* vol 19, no. 3, pp. 1154-1166, July 2004.

63. Muscettola, N., Nayak, P.,Pell, B.,and Williams, B., "Remote Agent: To Boldly Go Where No AI System Has Gone Before," *Artificial Intelligence,* vol. 100, 1997.

64. Hayden, Sandra C., Sweet, Adam J., and Christa, Scott E., "Livingstone Model-Based Diagnosis of Earth Observing One, AIAA-2004-6225, *AIAA 1st Intelligent Systems Technical Conference,* Chicago, IL, Sept. 20-22, 2004.

65. Aaseng, Gordon, Cavanaugh, Kevin, and Deb, Somnath, "An Intelligent Remote Monitoring Solution for the International Space Station, *IEEE Aerospace,* 2003.

66. Robinson, P., Shirley, M., Fletcher, D., Alena, R., Duncavage, D., Lee, C., "Applying Model-Based Reasoning to the FDIR of the Command & Data Handling Subsystem of the International Space Station," *iSAIRAS* 2003.

67. Hutcheson, R. and I.Y. Tumer. *Function-based design of a spacecraft power system diagnostics testbed.* in *ASME International Mechanical Engineering Congress and Exposition (IMECE).* 2005. Orlando, FL.
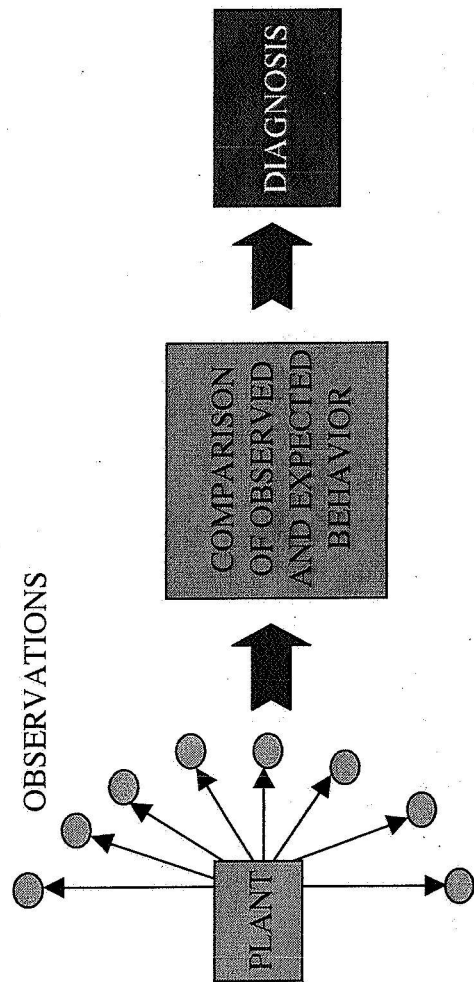
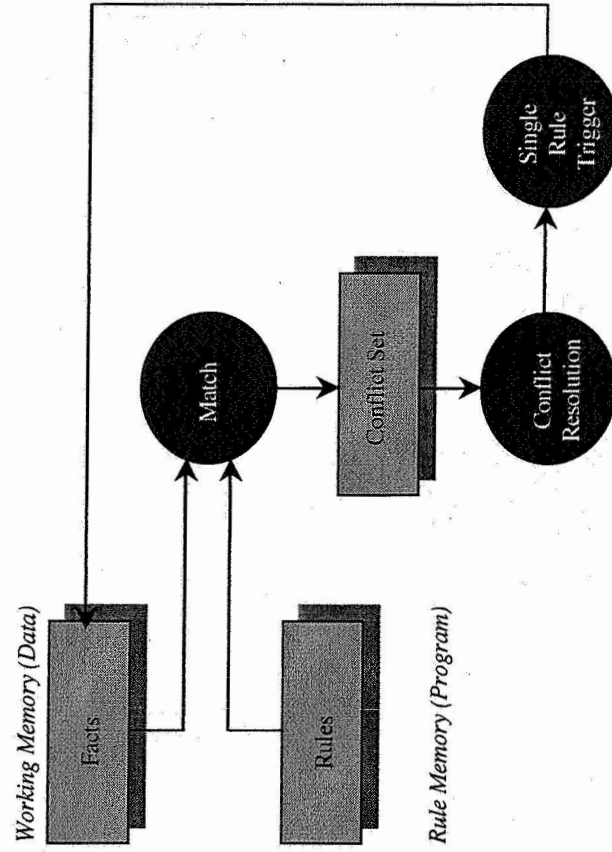# A Review of Diagnostic Techniques for ISHM Applications

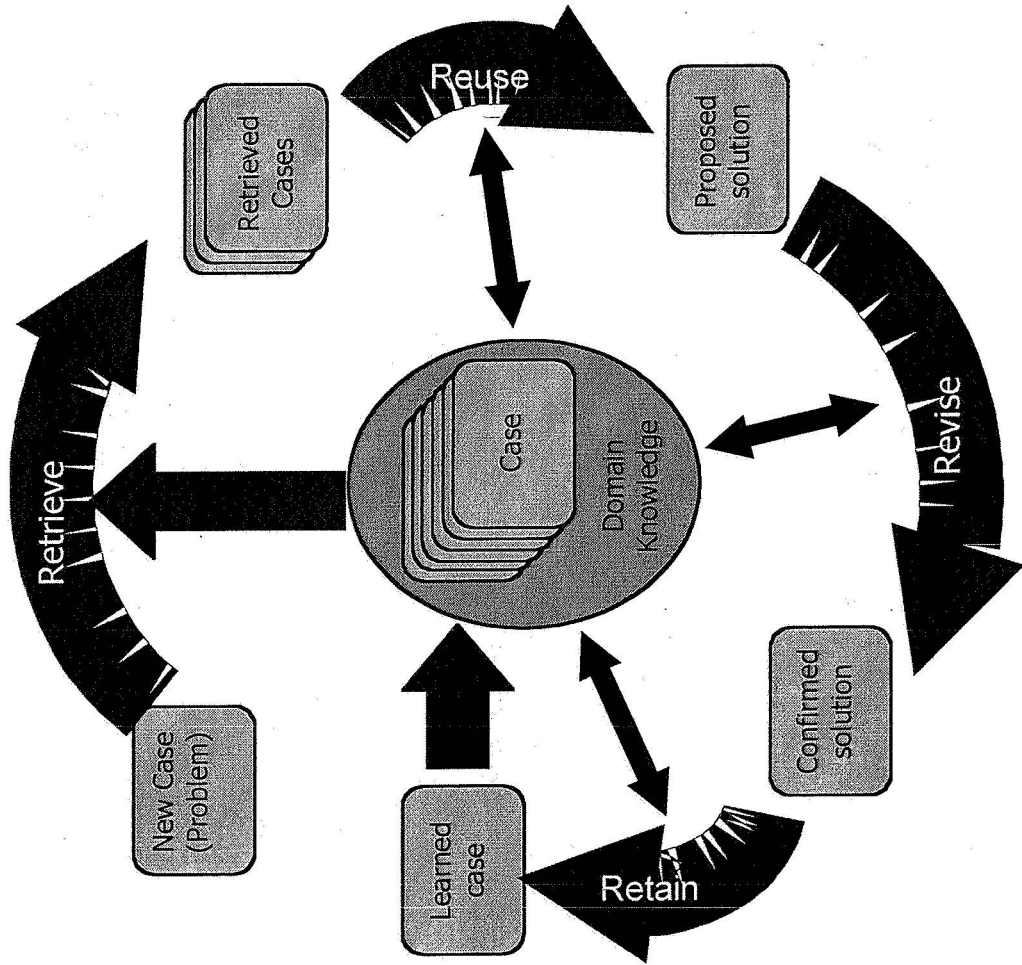Ann Patterson-Hine, Gordon Aaseng, Gautam Biswas, Sriram Narasimham, and Krishna Pattipati

A General Process for Diagnosis

OBSERVATIONS

PLANT

COMPARISON OF OBSERVED AND EXPECTED BEHAVIOR
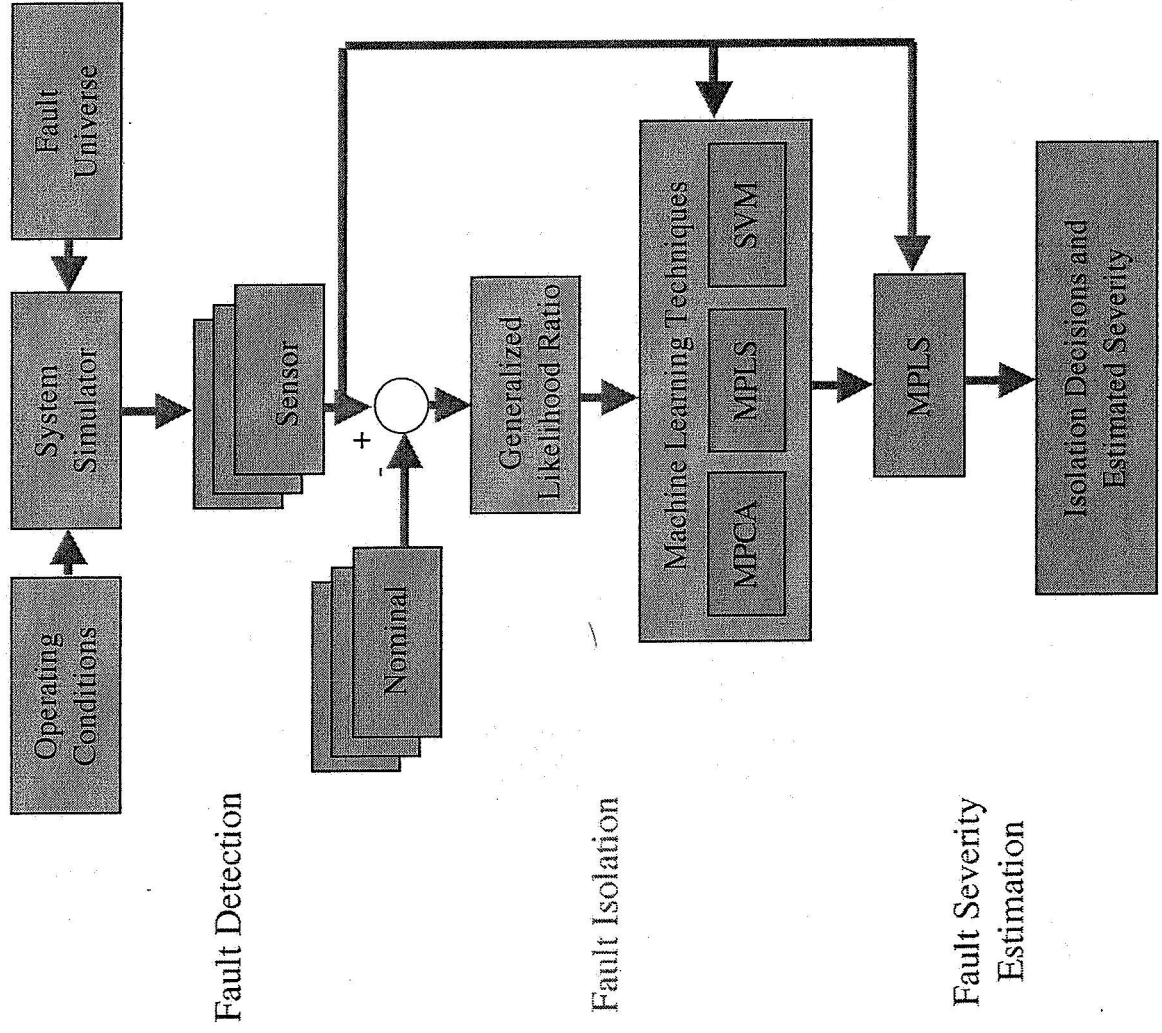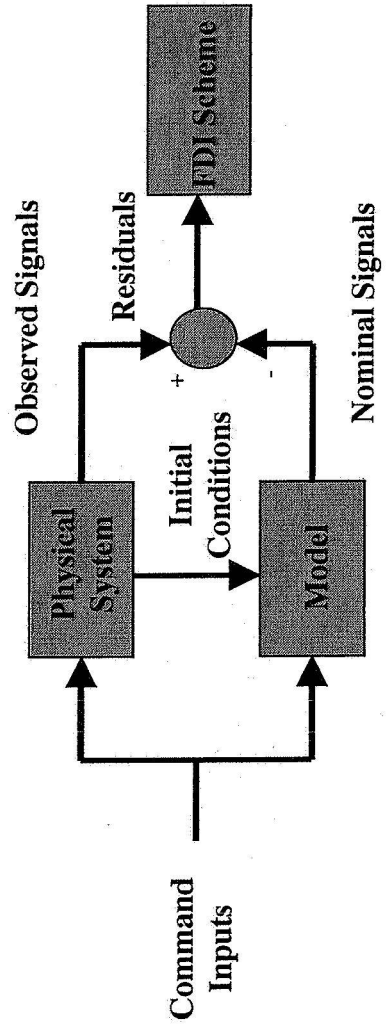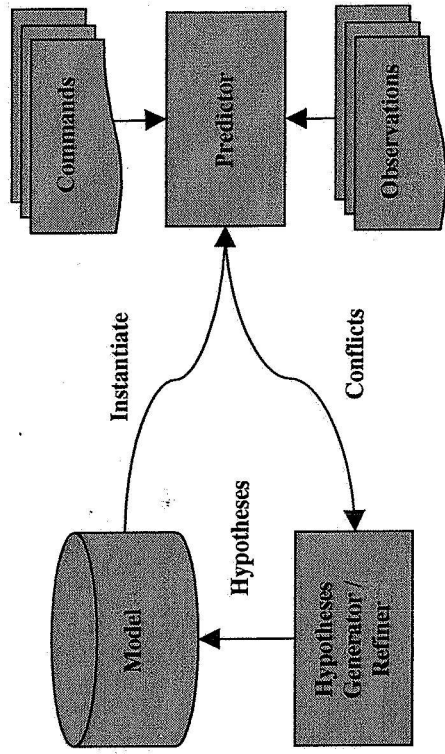
DIAGNOSIS

Rule-based Systems

Case-based Reasoning Systems

Learning Systems

Model-based Reasoning Systems

Automation Decision-Making

Deciding if detection and responses are automated, manual, or a collaborative effort requires processes that assess timing, criticality, technology maturity, reliability of HM and other factors

Crew Decisions, automated detection and annunciation

Automated with Crew Collaboration

Fully Automated

Onset Time (approx.)

>1 min

30.0 s

5.0 s

0 s

Impact of Failure

None

Marginal

Severe

Catastrophic