REPORT DOCUMENTATION PAGE	Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
27-08-2012	Final Report	12-Sep-2008 - 11-Sep-2010

4. TITLE AND SUBTITLE	
Dynamic Data-Driven Prognostics and Condition Monitoring of On-board Electronics (Phase-II, Final Report)	5a. CONTRACT NUMBER
	5b. GRANT NUMBER W911NF-08-C-0065
	5c. PROGRAM ELEMENT NUMBER 665502
6. AUTHORS Sudipto Ghoshal, Mohammad Azam, Sunil Dixit	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES	8. PERFORMING ORGANIZATION REPORT NUMBER
Qualtech Systems Inc Putnam Park, Suite 501 100 Great Meadow Road Wethersfield, CT 06109 -	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709-2211	10. SPONSOR/MONITOR'S ACRONYM(S) ARO 11. SPONSOR/MONITOR'S REPORT NUMBER(S) 54937-MA-ST2.2

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

Under This STTR project, Qualtech Systems Inc. (QSI), in collaboration with University of Maryland, and Northrop Grumman Co. (Aerospace systems) has developed a data-driven prognostic and condition monitoring solution that can be applied to complex electronic systems. The solution comprises of a library of detection, classification, forecasting, and inference algorithms implemented in software and hosted in QSI's TEAMS (Testability Analysis and Maintenance System) platform. The solution can detect, identify, and diagnose faults in

15. SUBJECT TERMS

Forecasting, Prognostics, STSA, Data Driven Prognostics, Time Series

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
UU	UU	UU	UU		Sudipto Ghoshal
					19b. TELEPHONE NUMBER 860-257-8014

Standard Form 298 (Rev 8/98)
Prescribed by ANSI Std. Z39.18

Report Title

Dynamic Data-Driven Prognostics and Condition Monitoring of On-board Electronics
(Phase-II, Final Report)

ABSTRACT

Under This STTR project, Qualtech Systems Inc. (QSI), in collaboration with University of Maryland, and Northrop Grumman Co. (Aerospace systems) has developed a data-driven prognostic and condition monitoring solution that can be applied to complex electronic systems. The solution comprises of a library of detection, classification, forecasting, and inference algorithms implemented in software and hosted in QSI's TEAMS (Testability Analysis and Maintenance System) platform. The solution can detect, identify, and diagnose faults in real-time; estimate degradation(s) in a system in real-time; estimate the time to fault, and estimate the remaining useful life (RUL) of system/subsystem/components. For the diagnostic (root cause identification) purpose, dependency model-based analysis is used in conjunction with data-driven methods. The solution has been tested on a small number of representative electronic systems; for instance: RUL estimation of computer components, and condition monitoring of KN-4073 INS/GPS (used on Army's MQ-8 Fire Scout UAV).

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

TOTAL:

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received Paper

TOTAL:

Number of Manuscripts:

Books

Received Paper

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	0.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:......	0.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:......	0.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):......	0.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:......	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:......	0.00

Names of Personnel receiving masters degrees

NAME
Total Number:

Names of personnel receiving PHDs

NAME
Total Number:

Names of other research staff

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Sub Contractors (DD882)

1 a. Northrop Grumman Corporation

1 b.

00000

Sub Contractor Numbers (c):

Patent Clause Number (d-1):

Patent Date (d-2):

Work Description (e): Aided QSI in generating data through KN-4073 Hardware in the loop simulation. Also provided ma

Sub Contract Award Date (f-1): 9/29/2008 12:00:00AM

Sub Contract Est Completion Date(f-2): 9/3/2010 12:00:00AM

---

1 a. University of Maryland

1 b. 3181 Glen L Martin Hall

College Park MD 20742

Sub Contractor Numbers (c):

Patent Clause Number (d-1):

Patent Date (d-2):

Work Description (e): Collaborated with QSI in developing STSA algorithms and their V&V

Sub Contract Award Date (f-1): 9/29/2008 12:00:00AM

Sub Contract Est Completion Date(f-2): 9/3/2010 12:00:00AM

Inventions (DD882)

Scientific Progress

Technology Transfer

STTR Topic A07-013: Dynamic Data-Driven Prognostics and Condition Monitoring of On-board Electronics

Phase-II, Final Report

Sep 10, 2010

Prepared under

Contract No. W911NF-08-C-0065
Contract Line Item Number (CLIN): 0001 & 0002
Period of Performance: Aug 29, 2008 – Aug 28, 2010
QSI Job no. SR1210

Issued by

US ARMY RDECOM ACQ-CTR – W911NF
4300 S. MIAMI BLVD.
DURHAM, NC 27703, USA
CODE: W911NF

Technical Point of Contact (TPOC): Dr. Harry Chang, Army Research Lab.

Contractor:	**Qualtech Systems, Inc. (QSI)**
Principal Investigator:	Sudipto Ghoshal, *sudipto@teamqsi.com*
Business Address:	100 Great Meadow Road, Suite 603
	Wethersfield, CT 06109-2378
Phone Number:	860 257-8014 x1101
Effective Date of Contract:	Aug 29, 2008

QSI Report No. *QSI-REP-10-042*

Distribution:	Bill Bratton, *bill.bratton@us.army.mil*
	reports@arl.army.mil
	Dr. Harry Chang, TPOC, Army Research Lab
	mouhsiung.chang@us.army.mil
	DCAA, *dcaa-fao2901@dcaa.mil*

DISTRIBUTION STATEMENT

Distribution and disclosure of the contents of this document are subject to FAR Clauses 52.227-11, 52.227-20, and DFARS 252.204-7000 incorporated in Contract no. W911NF-08-C-0065

TABLE OF CONTENTS

LIST OF FIGURES

List OF TABLES

1. Project Overview & Summary

Under This STTR project, Qualtech Systems Inc. (QSI), in collaboration with University of Maryland, and Northrop Grumman Co. (Aerospace systems) has developed a data-driven prognostic and condition monitoring solution that can be applied to complex electronic systems. The solution comprises of a library of detection, classification, forecasting, and inference algorithms implemented in software and hosted in QSI's TEAMS (Testability Analysis and Maintenance System) platform. The solution can detect, identify, and diagnose faults in real-time; estimate degradation(s) in a system in real-time; estimate the time to fault, and estimate the remaining useful life (RUL) of system/subsystem/components. For the diagnostic (root cause identification) purpose, dependency model-based analysis is used in conjunction with data-driven methods. The solution has been tested on a small number of representative electronic systems; for instance: RUL estimation of computer components, and condition monitoring of KN-4073 INS/GPS (used on Army's MQ-8 Fire Scout UAV).

2. Overall Scope and Background of the Project

Reliability of electronic systems is a prime determinant of efficacy of most modern day military hardware. With the increased complexity and utilization across all weapons systems ranging from soldier to aircraft to ships and land vehicles, the importance of electronics systems readiness has become crucial to mission success. Owing to the maturity in the arenas of material, control, communication and computer engineering, these electronic systems have become quite robust and surpass the level of reliability of most mechanical systems. However, like every other engineering systems, these systems also undergo degradation, and eventually experience faults and failure. The complexity of these systems, which provides unprecedented level of functional capabilities, puts forward formidable challenges in predicting, tracking and identifying the trend and source of degradations and failures. To overcome this challenge, it is necessary to look beyond the fault diagnostics and prognostics approaches that have been proven successful in mechanical systems; both from theoretical and application perspectives.

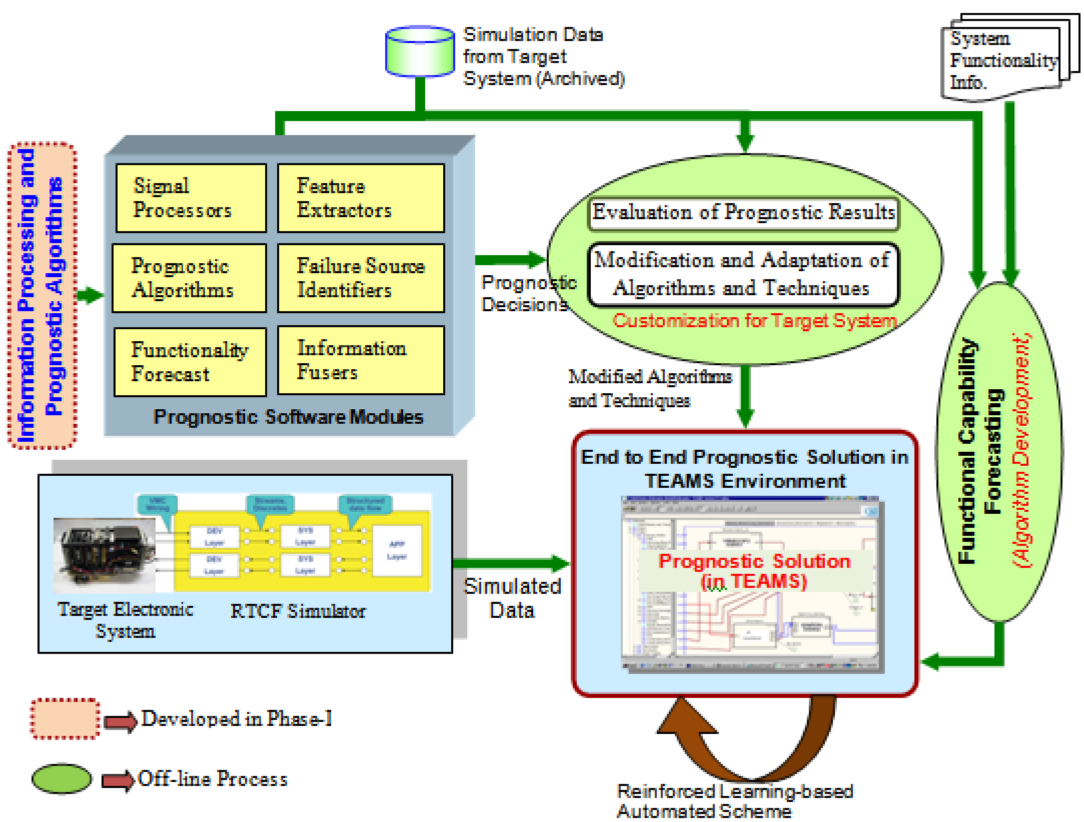

Figure 1: Phase-II Technology Concept

In Phase-I of this STTR effort, a team comprising of Qualtech Systems, Incorporated (QSI) and Center for Advanced Life Cycle Engineering (CALCE) at University of Maryland (UMD) developed a proof of concept approach to perform fault and degradation forecasting in electronic systems. In that effort, Northrop Grumman Corporation's (NGC) Integrated System, Western Region assumed the role of technology prime and provided voluntary guidance from its vast experience on the FCS program. In Phase-I, our team studied the trend of failures in electronic systems and investigated techniques to isolate and forecast them. As a result, we were able to develop an efficient system health, fault and degradation forecasting scheme using techniques ranging from signal processing to information fusion. The scheme was tested on a set of laptop computers (representative electronic systems) to validate its viability. Laptop computers house a wide range of electronic subsystems. Analogues or exact duplicates of those subsystems are used across the FCS platforms. Hence, we assume, upon customizing, the prognostic approach developed by the QSI-UMD team can be deployed to the onboard electronics system of real-world FCS platforms. Our team has identified the Guidance Navigation and Control (GN&C) system of Fire Scout, a Class IV UAV as the development and validation platform for the Phase-II effort. To obtain detailed knowledge and information on the target system, the QSI-UMD team has included NGC (Integrated Systems, Western Region) the prime contractor of Fire Scout as a subcontractor for the Phase-II effort. Since the GN&C is quite a large system, to satisfy the time limitation of the project, QSI-UMD-NGC team might selected the KN-4073 INS/GPS as the development and V&V platform for this Phase-II effort.

In the development of a deployable onboard prognostic solution for electronic systems, the major milestones are software implementation of algorithms, integration of the software modules into a seamless deployable/executable entity and rigorous testing and validation (on a test-bench or a prototype system) against a wide range of system faults/degradations. Figure 1 shows the proposed Phase II technology concept. To facilitate speedy integration and deployment, the capabilities of QSI's TEAMS design and analytic environment will be leveraged in this effort. The solution will be eventually deployed on NGC's hardware in the loop validation platform, known as "Real Time Component Framework" (RTCF) for testing and validation. NGC will also provide data obtained via hardware in loop (HIL) simulation of the target system(s) for algorithm testing, validating and refining in both pre and post deployment phases.

3. Phase-II Technical Objectives

The Phase-II effort targets to transform the data transformation, signal processing, detection and forecasting techniques developed through Phase-I effort into standardized software modules. Combining those modules into a end-to-end prognostic and condition monitoring solution is another integral target. Rigorous testing of the prognostic solution against a target system in an FCS validation environment so as to make it ready for eventual deployment on actual FCS platforms is the other major Phase-II objective. A Generalized prognostic and condition monitoring approach that facilitates fleet level forecasting and FCS platform health management can significantly improve the existing system maintenance paradigm in terms of effectiveness, time and cost. As a precursor to developing such a process, a Phase-II target has been set to investigate a software architecture capable of hosting a generalized prognostic and condition monitoring solution along with provisions of voluminous information ingestion and dissemination. Enriching the library of prognostic and condition monitoring algorithm library with new techniques and expanding their capabilities to perform forecast on functional requirement fulfillment capability constitute an additional Phase-II technical objective. Specific technical objectives for the Phase-II are:

1. Development of a library of predictive software modules and expand it via introduction of new techniques
2. Development of a test-bench deployable version of the prognostic solution
3. Verification and validation of the prognostic software on the target system(s) in an FCS simulation environment using system model(s) and hardware in-the-loop simulation

4. Investigation on a suitable software architecture to facilitate automated data collection, data integration, transformation, prediction and dissemination of prognostic information for a fleet of FCS platforms

The Phase-II effort will be jointly pursued by QSI, UMD and NGC. Involvement of NGC, the prime contractor of the Fire Scout UAV ensures that the outcome prognostic solution from this effort will satisfy the technological requirements for its eventual transition to the real-world FCS platforms.

4. Phase-II Technical Accomplishments & Task Results

Through this Phase-II effort the QSI team developed and matured a suite of degradation tracking and forecasting techniques applicable to complex electronic systems. The techniques were incorporated into TEAMS-RTX*, the analytical algorithm implementation, embedding and real-time reasoning tool of TEAMS. Performance degradation experiments were conducted on a KN-4073 INS/GPS system, and a large set of data was collected. Using this data and data available from public sources, the QSI team tested and validated the TEAMS-RTX based prognostic solution.

4.1. Technical Results

Detailed account of accomplishments and task progress achieved during the Nov, 09 – Feb, 10 period follows

4.2. TASK-1: SOFTWARE IMPLEMENTATION AND ENRICHMENT OF ALGORITHMS

The overall goal of Task-1is to provide a range of a software-implemented analytic modules that can be integrated into a complete prognostic and condition monitoring solution for onboard electronic systems of FCS platforms. Apart from high degree of analytic efficacy, these modules also need to satisfy the portability (across different OS), adaptability (easy embeddability in diverse platforms), and compatibility with different open system architectures (such as OSA-CBM, CLOE [2], PS-MRS [3], etc). During the Nov09 - Feb 09 period, the QSI team worked on completion of implementation of a STSA scheme techniques into C dll files that can be plugged into QSI's TEAMS-RTX based analytic environment. Work on enhancement of the ARX algorithm via utilizing RLS regression [27] was also completed. Descriptions of the work done under this task during the recent period of performance are provided in subsections 4.2.2, 4.2.3, and 4.2.4.

4.2.1 Research on Open System Architectures

This subsection presents information about the open system architecture requirements for condition based maintenance. Machinery Information Management Open Systems Alliance (MIMOSA) a non-profit organization has released a document on open system architecture on condition-based maintenance [8]. The standard is written in the Unified Modeling Language (UML). The basis for this report is the OSA-CBM primer [8]. This primer as well as the OSA-CBM specifications itself, are built on ISO 13374, which provides a uniform method for implementing an infrastructure that integrates data from different sources in a system [7]. OSA-CBM specifies application programming interfaces (API) for target platforms, such as OMG CORBA, COM/DCOM, XML, and CIP [22]. The benefits of OSA-CBM include reduced cost, better specialization/competition, cooperation, and the ability to write proprietary algorithms. ISO has also released information on condition monitoring and diagnostics of machines specifically dealing with data processing, and communication [11], [12].

Open system architecture has been identified as the best way to allow different systems or frameworks to interact with each other. One definition for open system architecture is the ability to provide an environment that allows data/information to be interfaced and shared with remote devices and for the user to enhance or replace any internal functions. Another definition is that open system architecture is a

* TEAMS-RTX is an extended functionality version of TEAMS-RT® and it has been developed via QSI's internal R&D project. Several DoD and NASA SBIRs contributed towards addition of features to TEAMS-RTX.

framework in which many of the interfaces conform to widely agreed standards so that evolution of the system and interoperability can be achieved. Requirements for open system architecture are that they should be based on industry standards and that the architecture and the standards must be interdependent.

Examples of open system architecture would be the CLOE and PS-MRS programs. CLOE or Combat Logistics Operating Environment is an integrated system designed to use information from self-diagnosing and self-reporting vehicles to interact with a network sustainment infrastructure that supports condition based maintenance. Current information is used to anticipate logistics functions, combat readiness, and survivability of the vehicle or system of interest. Thus, it allows for a system to be sustained through prognostic logistics. The condition of the unit, the environment it operates in, and its ability to perform a task on data that is gathered in real time by the system itself, which is then used to determine the course of action for the unit. PS-MRS, or Platform Soldier-Mission Readiness Systems is a software that serves the function of a knowledge source to enhance mission planning and logistics operations. Much like CLOE, its purpose is to integrate different levels of a decision making process into one fluid body of knowledge.

The input and output data formatting for the algorithms can be stored as comma separated variables (.csv) files. These csv files are considered open system and can be read by any program currently implemented on military systems. Being open system these are also compatible as inputs for with software developed by other organizations. The QSI-CALCE team can adjust the outputs and inputs for the algorithms to match the requirements for the FCS platform. Using .csv files the programs developed using the C++ software import and export from most databases types, thereby providing data compatible analysis module within the PHM architecture, across all platforms. CSV files are really important for this program because of the lack of hard drive space. Unless the data will be transmitted to a remote site that has the ability to store all the data than CSV files will have to be used. XML files of the same data can be up to 13 times larger than the same data in CSV form [13]. XML is a much more structured way of storing data and easier to modify but the file size are increased tremendously due to the extra XML tags in the file. To help understand the flow of a prognostics system, ISO came up with a block diagram to illustrate the major components. Each block serves a general purpose in prognostics solution. Figure 1 is the block diagram from ISO 13374-1:2003. According to ISO the display should include the last four blocks. Utilization of this architecture will lead to a standard way of displaying data and less confusion on the user.

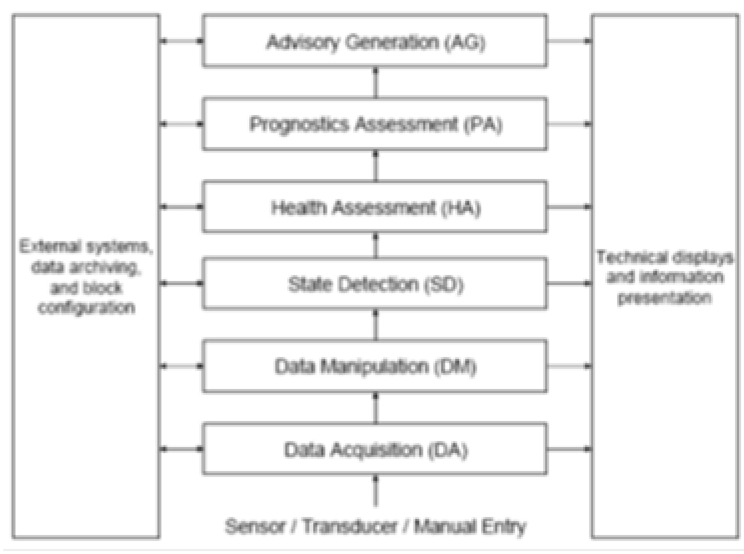

Figure 2: Data Processing and Information Flow as Specified by ISO 13374 [11]

4.2.1.1 Functional Blocks and Types of Information

There are six functional blocks of the CBM, as well as interfaces between them. The standard allows for integration of these separate parts by specifying inputs/outputs of the components. The six components are as follows: 1) Advisory generation (AG), 2) Prognostics assessment (PA), 3) Health assessment (HA), 4) State detection (SD), 5) Data manipulation (DM), and 6) Data acquisition (DA). Dunsdon and Harrington [7] addressed two supplemental functional blocks in their implementation of OSA-CBM: 1) Maintenance action and 2) Presentation/GUI (PR).

These blocks specify three types of information: 1) Data, or information and event sets that a block has generated, 2) Configuration, which describes the module's input sources, algorithms that process input, outputs, and output specifics, and 3) Explanation, describes the data used to produce an output.

ISO 13374 specifies the six functional blocks of CBM and their inputs/outputs. MIMOSA's OSA-CBM is an implementation of that standard such that it adds data structures and interface methods for the six functional blocks. When implementing OSA-CBM, middleware communication must be specified (XML, CORBA, RMI, DCOM) since it is not specified by the standard. Below is a figure specified by ISO 13374 technical documentation.

Although these blocks appear in a hierarchy, all are free to communicate as long as the proper URL or location is known. The OSA-CBM primer [8] provides the following information based on the six functional blocks. Functionality and characteristics of these blocks are listed below:

1. Data Acquisition (DA)

- Transforms output from a sensor into a scaled digital format
- Collects analog, digital, and manual data; converts analog data to digital data
- The output of DA blocks should contain digital data, time-ordered data, and data quality indication

Functionality of the DA block is illustrated through an example in Figure 3.

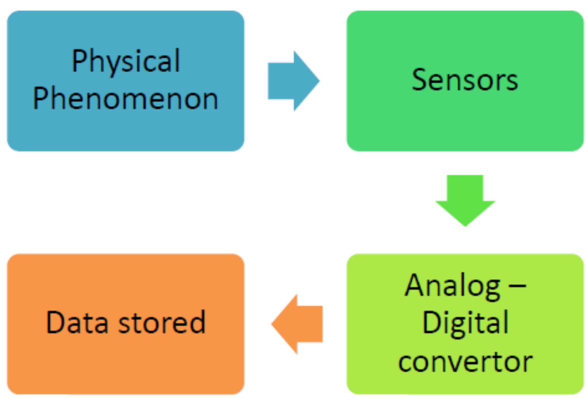

Figure 3: Example of Data Acquisition System

2. Data Manipulation (DM)

- Digital data from the DA block is converted into a desired form that characterizes features of interest for system condition monitoring and diagnosis
- Provides signal analysis, descriptors, and virtual sensor readings from raw data
- DM can include specialty functions such as Fast Fourier Transforms or wavelets
- Some of the DM outputs include extracted features, filtering, normalization, and calculated (non-interpretative) values

Functionality of the DM block is illustrated through an example in Figure 4.

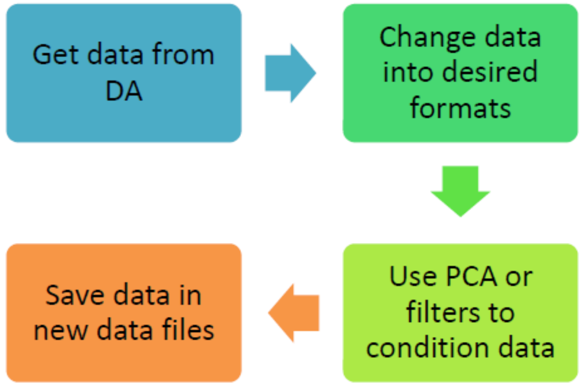

Figure 4: Flow of DM

3. State Detection (SD)

- Compares (current and historical) outputs from the DA and DM blocks against expected baseline profiles or operational limits
- The SD block provides indicators that can be used by the Health Assessment block for alerts
- Outputs from this block can include indicators of values exceeding the threshold, severe deviation from the threshold, rate of change, and statistical distributions

Functionality of the SD block is illustrated in Figure 5.

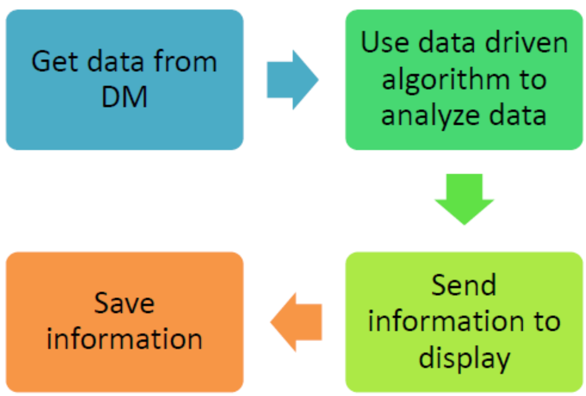

Figure 5: Flow for SD

4. Health Assessment (HA)

- By tailoring the algorithms in DA, DM, and SD, the HA block will be able to determine the state of health and potential failures, along with their likelihood probability
- Generates recommendations, evidence, and explanation
- The HA block can also detail evidence for the diagnosis or health grade

5. Prognostics Assessment (PA)

- Predicts future health and failure modes depending on the current health state and expected future loading conditions
- Predicts remaining useful life
- Typical output of a PA block details future health grade and failure events of the system, associated likelihood probability, and possible explanation

6. Advisory Generation (AG)

- Provides information regarding maintenance or configuration changes for optimization of remaining system life
- Integrates information from all functional blocks within the software to provide useful information to the user

Displaying the data and health of the system can be just as important as recording the data from the system. The display needs to be well laid out with the most important information on display. The display also needs to explain the state of the system clearly and give general advice for the actions that may need to be taken. Figure 6 is an example from ISO 13374-1:2003. The recommended display is made up of five areas. The areas are general and allow for some changes so that the display is right for prognostics solution.

- ***Area 5***

This area is where the system being monitored is identified. This is important because the user will want to make sure systems do not get confused with one another.

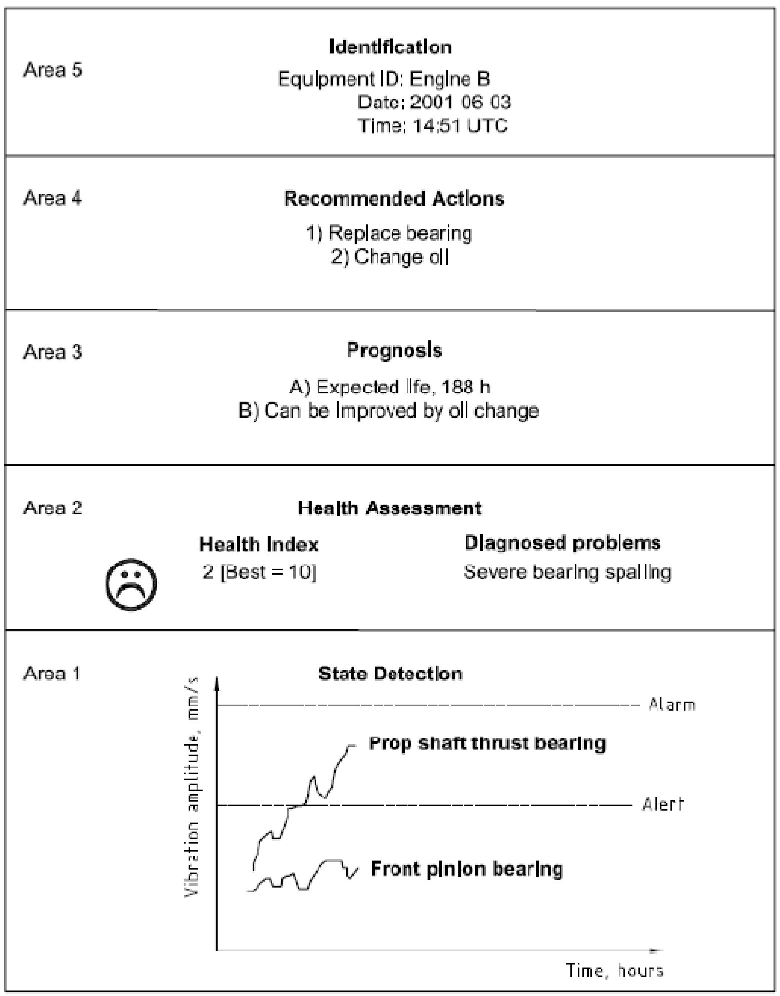

Figure 6: Display for Engine in ISO 13374-1:2003 [11]

- ***Area 4***

This area is where the advice for the system is displayed. The type of advice depends on the severity of damage in the system.

- ***Area 3***

This area is for the prognostics of the system. This should clearly display the amount of life remaining in the system is. This should have the ability to change depending on the maintenance and actions taken by the user to extend the life of the system.

- ***Area 2***

This area gives a general overview of the system. It should include a health index based on a 0 to 10 where 0 is a failure in the system. This area should also include the failure that the system is experiencing.

- ***Area 1***

This area is used to display some of the data that is being recorded. This is normally data taken over time. This allows the user to see trends over time and possibly see abnormities in the data. It should also include target values that might signal a fault or failure in the system.

4.2.1.2 Functions

As part of MIMOSA's implementation of ISO 13374, the OSA-CBM specification has 14 functions to be implemented by the different layers.

Table 1: Methods for blocks as specified by the OSA-CBM Primer [8]

Method	Description	Input	Return Type
epRequestDataEvent	Returns a specified Data Event for the block	MonitorId m	DataEvent
epRequestDataEvent	Returns a specified Data Event Set for the block	MonitorIdList mList	DataEventSet
epGetRequestDataEventSetStatus	Gets the status of a data event request	int RequestID	double
epGetRequestDataEventStatus	Gets the status of a data event request	int RequestID	double
epRequestConfig	Returns a configuration class that provides the configuration information type	ConfigRequest configRequest	Configuration
epRequestExplanationDataSet	Provides actual Data Event Set used for calculation	MonitorIdList mList	ExplanationDataSet
epRequestExplanationDataRefSet	Provides a handle to a well-known location where data is stored	MonitorIdList mList	ExplanationDataRefSet
epRequestExplanationSrcs	Provides pointers to data	MonitorIdList mList	ExplanationSrcs
epRequestExplanationSrcsStr	Provides pointers to data as a string	MonitorIdList mList	ExplanationSrcsStr
epNotifyControl	Allows for changes to module control parameters on the fly	ControlChange controlChange	void
epRequestControl	Returns module's current control parameters	ControlInfoRequest controlInfoRequest	ControlInfo
epNotifyApp	Sets application specific information	AppRequest appRequest	void
epRequestApp	Returns application specific information	AppRequest appRequest	AppRequestRtn
epRequestErr	Returns and error generated by the model	ErrorRequest errorRequest	ErrorNotify

4.2.1.3 Communication

OSA-CBM specifies four types of communication, which are shown in the following figure. The user must choose the technology and type of communication for implementation.

4.2.1.4 Programming

Programming and system to system communication is and will be extremely important for a successful prognostics solution. Each prognostics solution may be unique but their ability to communicate to each other and to the military's computer systems will be critical. There are many different programming paradigms that could be used in a prognostics solution but some may lend themselves better for a certain outcome.

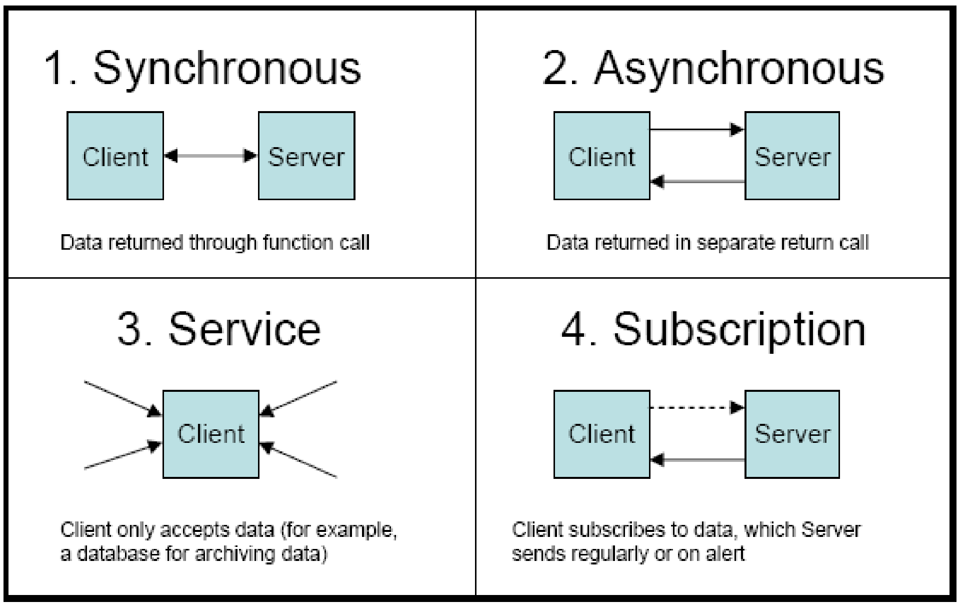

Figure 7: Types of Communication [8]

C, C++ and C# are the most potential candidate languages in realization of the open systems architecture. The major reasons behind the choice lie in proliferation, standardization, and ultra-wide range of usability. C, and C++ are standardized by ISO/IEC 14882:2003. C# is standardized by European Computer Manufacturers Association (ECMA-334) [14] and ISO (ISO/IEC 23270) [16]. C# is an incremental enhancement of the C++ programming language and more directly reflects the underlying Common Language Infrastructure (CLI). CLI is an open specification published under ECMA-335 [15] and ISO/IEC 23271 [17]. C# is developed by Microsoft for the .NET framework. C# is intended for use in developing software components for deployment in distributed environments. C++ is one of the most viable choice, both from viewpoint of functionality and accessibility; it is an open language unlike Java or Visual meaning that it is also free. It is also one of the most popular programming languages made. It is a multi-paradigm programming language allowing many programming styles. C++ and C# allows the programmer to do almost any type of coding including object oriented programming. C and C++ are widely accepted language across many vendors. Also programs written in C++ and C# are able to run without the use of a virtual machine like Java.

4.2.1.5 Implementation

For building of an OSA-CBM system, the primer documentation [7] recommends the following steps:

1. Choose a middleware technology (DCOM, CORBA, Web Services, Java RMI, etc.).
2. Transform OSA-CBM UML into a format compatible with the chosen technology.

3. Choose a communication type (synchronous, asynchronous, service, or subscription).
4. Build OSA-CBM compliant interfaces for each functionality block.
5. Build information processing modules for each functionality block.
6. Combine interfaces and information processing modules as a working system.

4.2.2 Implementation of Signal Processing, Diagnostic, Prognostic, and Information Fusion Techniques into Generic and Portable Software Modules

In Phase-I of this STTR effort a vast collection of signal processing, feature extraction, forecasting, classification and information fusion techniques were investigated; a portion of those were also implemented in software for studying their feasibility and efficacy towards achieving the goal of this effort. The algorithms that were implemented in MATLAB in Phase-I were converted to C dlls in the first quarter of Phase-II, while some additional algorithms were implemented during the third quarter of Phase-II. These C modules are being used as plug-ins of the TEAMS-based prognostic solution (under development, see section 4.3 for details). Hence, while performing the conversion, we ensured that the resultant modules satisfy the API requirements of the aforementioned solution.

So far, the following algorithms have been converted/implemented as C dlls

Cumulative Sum (CUSUM) test

Fourier Transform

Wavelet Transform Framework: The framework is able to work with different types of wavelets and window functions†.

Principal Component Analysis (2-D PCA)

Fisher's Discriminant Analysis (both Linear and Quadratic)

Support Vector Machine (classification only)

Auto regressive

Auto Regressive Moving Average analysis (ARMA)

Auto Regressive with Exogenous Input (ARX), and ARMAX

Kalman filter

General Algebraic and Statistical Functions

During the Aug 09 – Nov 09 period of performance, QSI worked on implementation/conversion of HMM state propagation and decoding algorithms. The strategy followed in this task is to prioritize the conversion of a set of algorithms that suffice the requirements for implementing a testable end-to-end prognostic solution. At this point of time, sufficient number of algorithms has already been converted to pluggable dll form. Therefore, in the upcoming period of performance QSI will concentrate more on verifying the accuracy of the algorithms regarding fault forecasting.

4.2.3 Optimization of ARX Algorithm Performance

The ARX algorithm can be implemented with different types of regression techniques. The underlying regression technique is a major determinant of ARX performance in terms of both training cost and accuracy. Autoregressive time series forecasting techniques plays a major role in the data-driven prognostic scheme being implemented by the QSI team. Therefore, during the Aug 09 – Nov 09 period of performance, we worked on implementation of ARX using a customized recursive least square algorithm. The details of the customization are stated below.

† DB, Har, Sym, and Coiflet types of wavelets have been worked on

ARX Model:

- Linear difference equation:

$$\mathbf{y}(\mathrm{n}) + \mathrm{a}_1\mathbf{y}(\mathrm{n}-1) + \cdots + \mathrm{a}_{\mathrm{n_a}}\mathbf{y}(\mathrm{n}-n_a) = \mathrm{b}_1\mathbf{u}(\mathrm{n}-1) + \cdots + \mathrm{b}_{\mathrm{n_b}}\mathbf{u}(\mathrm{n}-n_b) + \mathbf{e}(\mathrm{n})$$

- Parameter Vector:

$$\boldsymbol{\theta} = [\mathrm{a}_1 \;\; \mathrm{a}_2 \;\; \ldots \;\; \mathrm{a}_{n_a} \quad \mathrm{b}_1 \;\; \mathrm{b}_2 \;\; \ldots \;\; \mathrm{b}_{n_b}]^{\mathrm{T}}$$

- Output/Input data vector:

$$\boldsymbol{\varphi}(\mathbf{t}) = [-\mathbf{y}(\mathrm{n}-1) \;\; \ldots \; -\mathbf{y}(\mathrm{n}-n_a) \;\; \mathbf{u}(\mathrm{n}-1) \ldots \;\; \mathbf{u}(\mathrm{n}-n_b)]^{\mathbf{T}}$$

Consequently, the equation can be reduced down to the "scalar product of the know data vector $\boldsymbol{\varphi}(\mathrm{t})$ and the parameter vector $\boldsymbol{\theta}$ [26].

$$\hat{\mathbf{y}}(\mathrm{n}|\boldsymbol{\theta}) = \boldsymbol{\varphi}(\mathrm{n})^{\mathbf{T}}\boldsymbol{\theta} + \boldsymbol{\mu}(\mathrm{n})$$

Recursive Least Squares:

- Update equations:

$$\widehat{\boldsymbol{\theta}}(\mathrm{n}) = \widehat{\boldsymbol{\theta}}(\mathrm{n}-1) + \mathbf{g}(\mathrm{n})\big[\mathbf{y}(\mathrm{n}) - \boldsymbol{\varphi}^{\mathbf{T}}(\mathrm{n})\widehat{\boldsymbol{\theta}}(\mathrm{n}-1)\big]$$

$$\mathbf{g}(\mathrm{n}) = \frac{\mathbf{P}(\mathrm{n}-1)\boldsymbol{\varphi}(\mathrm{n})}{\boldsymbol{\lambda}(\mathrm{n}) + \boldsymbol{\varphi}^{\mathbf{T}}(\mathrm{n})\mathbf{P}(\mathrm{n}-1)\boldsymbol{\varphi}(\mathrm{n})}$$

$$\mathbf{P}(\mathrm{n}) = \frac{\mathbf{P}(\mathrm{n}-1) - \dfrac{\mathbf{P}(\mathrm{n}-1)\boldsymbol{\varphi}(\mathrm{n})\boldsymbol{\varphi}^{\mathbf{T}}(\mathrm{n})\mathbf{P}(\mathrm{n}-1)}{\boldsymbol{\lambda}(\mathrm{n}) + \boldsymbol{\varphi}^{\mathbf{T}}(\mathrm{t})\mathbf{P}(\mathrm{n}-1)\boldsymbol{\varphi}(\mathrm{n})}}{\boldsymbol{\lambda}(\mathrm{n})}$$

- Initialize:
  - Initial conditions for both the parameter vector, θ, and the inverse correlation matrix, $\mathbf{P}(n)$, are needed.
  - This can be done one of two ways:
    i. Recursively "build up" the matrix until it is full rank using eq. (e.1) below and compute w(0) from $\mathbf{P}(0)$, and the cross-correlation vector using eq. (e.2):

$$\mathbf{P}(0) = \left[\sum_{i=-\mathrm{p}}^{0} \boldsymbol{\lambda}^{-i}\boldsymbol{\varphi}(i)\boldsymbol{\varphi}^{\mathrm{T}}(i)\right]^{-\mathbf{1}} \qquad (\mathrm{e.1})$$

$$\mathbf{r}(0) = \sum_{i=-p}^{0} \boldsymbol{\lambda}^{-i}\mathbf{y}(i)\boldsymbol{\varphi}(i) \qquad (\mathrm{e.2})$$

$$\boldsymbol{\theta}(0) = \mathbf{P}(0)\mathbf{r}(0) \qquad (\mathrm{e.3})$$

Where, p is the filter order (size of data vector) and λ is assumed to be constant. This approach minimizes the weighted least squares error. Although a disadvantage of this approach is that it requires direct inversion of the autocorrelation matrix and there is an extra delay of p+1 samples before coefficients can be calculated and updated.

    ii. Second approach is to assume the autocorrelation matrix and parameter vector are initialized as follows:

$$\mathbf{R}(0) = \; \boldsymbol{\delta}\mathbf{I}$$

$$\mathbf{P}(0) \triangleq \; \boldsymbol{\delta}^{-\mathbf{1}}\underline{\underline{\mathbf{I}}}$$, and

$$\boldsymbol{\theta}(0) = \underline{\mathbf{0}}$$

Where, δ, is a small positive constant and **I** is the identity matrix. This method is computationally more efficient compared to the last method, because no matrix inversion is necessary and there is no time delay for parameter estimation. The disadvantage of this method is that it "introduces a bias in the least squares solution" [27], although, "the bias goes to zero as n increases" [27].

ALGORITHM:	**Recursive Least Squares**
Parameters:	*p = Filter order* *λ = Exponential weighting factor* *δ = Small value to initialize inverse autocorrelation matrix*
Initialization:	$\theta(0)$ $P(0)$
Computation:	For n = 1, … $z(n) = P(n-1)\varphi(n)$ $g(n) = \frac{z(n)}{\lambda+\varphi^T(n)z(n)}$ $\alpha(n) = y(n) - \varphi^T(n)\theta(n-1)$ $\theta(n) = \theta(n-1) + \alpha(n)g(n)$ $P(n) = \frac{1}{\lambda}[P(n-1) - g(n)z^H(n)]$

*Note: $z^H(n) = \varphi^T(n)P(n-1)$

QSI is in the process of incorporating the algorithm under the ARX algorithm plugin for the TEAMS-RTX analytic environment. Its incorporation and testing will be completed during the upcoming period of performance.

4.2.4 Implementation of the STSA as TEAMS-RTX Pluggable Scheme

In Phase-I the QSI-UMD team developed a scheme for utilizing STSA for generating diagnostic and prognostic measures and indicators (see Figure 8). During the Nov 09 - Feb10 QSI completed the implementation of most of the individual algorithms of the scheme as RTX pluggable components. The data preprocessing component comprises of PCA and Mahalanobis Distance options; several wavelet transform options including Dubechies, Coiflet, Symlet, and Haar are avialble, standard Shannon Entropy computation process was used in stacking, partitioning and generating symbolic time series. A straight-forward Markov model was generated from the time series. QSI continued working on optimizing the Markov model under Task-4.

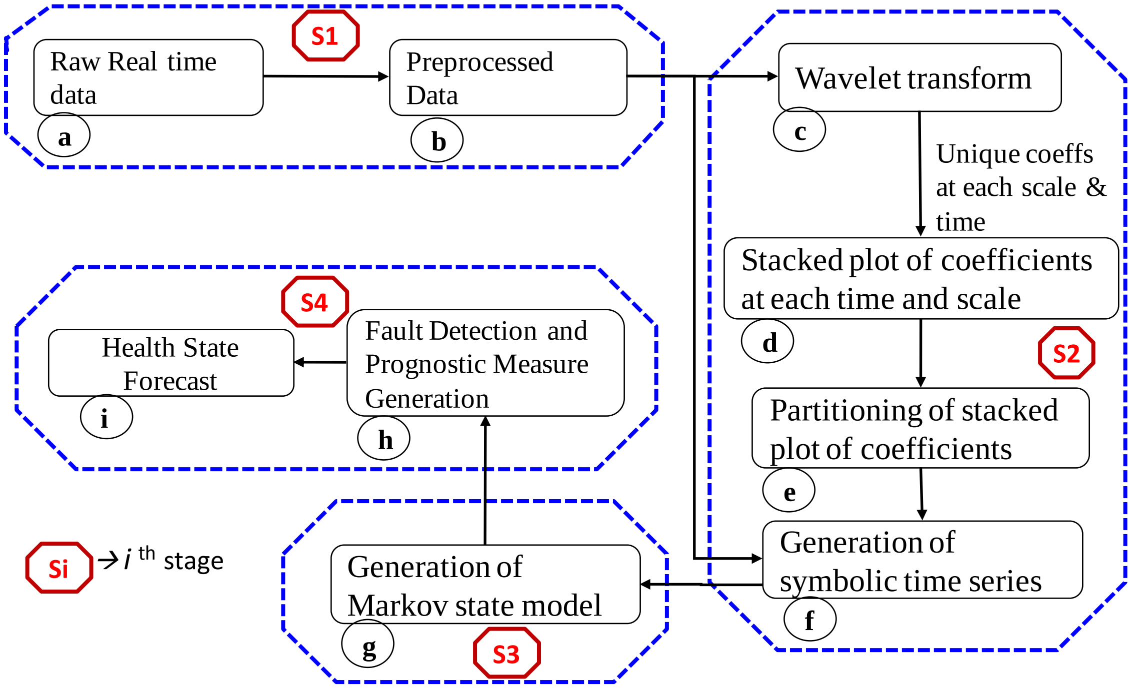

Figure 8: QSI-UMD STSA Scheme

4.2.5 Investigation on Optimization Algorithms

A part of Task-1 focuses towards investigation of optimization algorithms that can be used in data-driven prognostics and condition monitoring systems. The objective of this subtask is to develop techniques that ensure optimal utilization of sensors (through analyzing the worth of monitored data in health and performance forecasting) and selection of the most appropriate set of algorithm (so as to maximally exploit the information content of data, historical results, and system characteristics) for accurate condition monitoring and forecasting. During the June, 09 – Aug, 09 period of performance, scopes of optimization techniques for prioritization of parameters, as well as in STSA were investigated. A short description of the work is given in the following subsection. This is an ongoing effort and more algorithms will be evaluated in year 2 of the project.

4.2.5.1 Use of Optimization for Prioritization of Parameters

While collecting parametric data from an electronic system for health monitoring, the data might have been accessed from system output signals and/or from internal sensors. Electronic systems are using more sensors than ever. With the variety of sensor types and number of sensors on a system using the right data for the right purposes can often be confusing. It is imperative to use data from the right sensors so that proper conclusions can be made about the system. Without proper analysis of the sensor data and the condition of the system, the correct sensors may never be chosen.

Principal Component Analysis (PCA) and Partial Least Squares (PLS) are two efficient techniques for identifying the sensors that provides maximally useful information in fault detection, identification and prognostics [18], [19]. Both of these techniques can rank order the sensors (monitored parameters) in terms of their contribution in the variability of the observed data set. It's a verified fact that variability is the major indicator of state transition in a system. Such system state can represent health condition, performance, operating mode, loading conditions, etc. Genetic algorithms [20] are yet another potential technique for identification of the most effective sensor set for fault detection, identification and prognostics. During the upcoming period of performance, the QSI-CALCE team will perform experiments on the above mentioned techniques to evaluate their worth in prognostics and health management of onboard electronic systems.

4.2.5.2 Use of Optimization Techniques in STSA

There are several ways in which optimization techniques can be used in the symbolic time series analysis. For instance, use of Mahalanobis distance (MD) [21] instead of Euclidean distance creates a better identification of the data distribution, as shown in Figure 9. While point A is further from the origin than point B, it is more in line with the Mahalanobis definition of a healthy system, and therefore point B is the one that represents an anomaly. On the overall, Mahalanobis distance also represents that data better. MD represents the data better because MD can take into account the directions of the data.

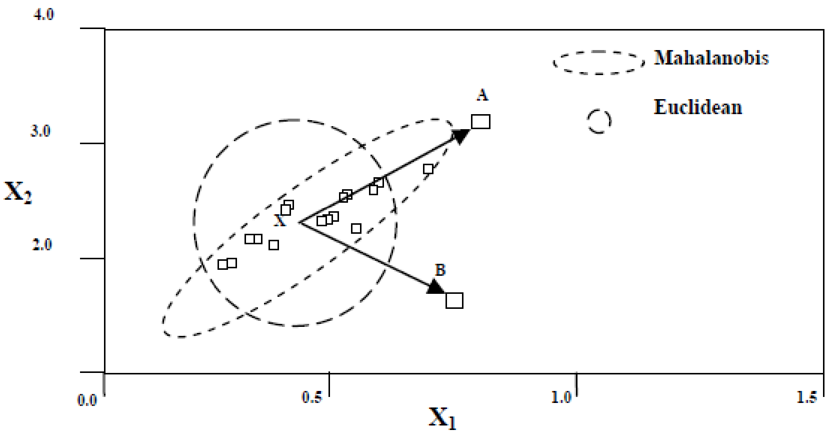

Figure 9: Effect of using Mahalanobis vs. Uclidean Distance in STSA

STSA has built-in loops for optimization for coefficients and partitions. This ensures that each partition contains an optimal number of symbols for the data. STSA starts with a random number or user defined number of coefficients and partitions. As the program is run, this number is updated and corrected to provide the optimal number of partitions and coefficients without any user input. The number of symbols to select as a Markov state is also optimized. This is done by maximizing the Shannon Entropy as mentioned earlier. This is important because if the wrong number of states is chosen then the number of possibilities of states may be too great or too low. Another reason for the correct number of Markov states is that some of the states may never be achieved and thus some states will be wasted. Maximizing the Shannon Entropy finds the optimal number of Markov states. Scopes for utilizing optimization techniques within the STSA scheme (developed during Phase-I), will be further investigated during the upcoming period of performance.

4.3. Task-2: Development of TEAMS-based Solution for Automated Data-driven Prognostics

A data-driven prognostic solution comprises of a collection of forecasting, fault detection and isolation and information processing techniques. Different functional units of the diagnostic-prognostic solution (e.g., forecasting engine, tests, dependency models, real-time processing engine, external plug-in modules, etc) and the information interchange systems (for importing monitored data and exporting diagnostic/prognostic decisions) need to work concertedly for accurate assessment of system health status and performing health forecasts. To ensure such concerted functionality, the prognostic solution should be built within a framework that defines the connectivity, communication and execution processes and/or protocols of the functional units in the system. Additionally, the framework should have the capability to facilitate designing and embedding of dependency models, induct analytic techniques and verify the efficacy of a diagnostic/prognostic and condition monitoring solution using multi-criteria analysis (e.g., remaining useful life, reliability, testability, etc). A software framework for such a solution is shown in **Figure 10**.

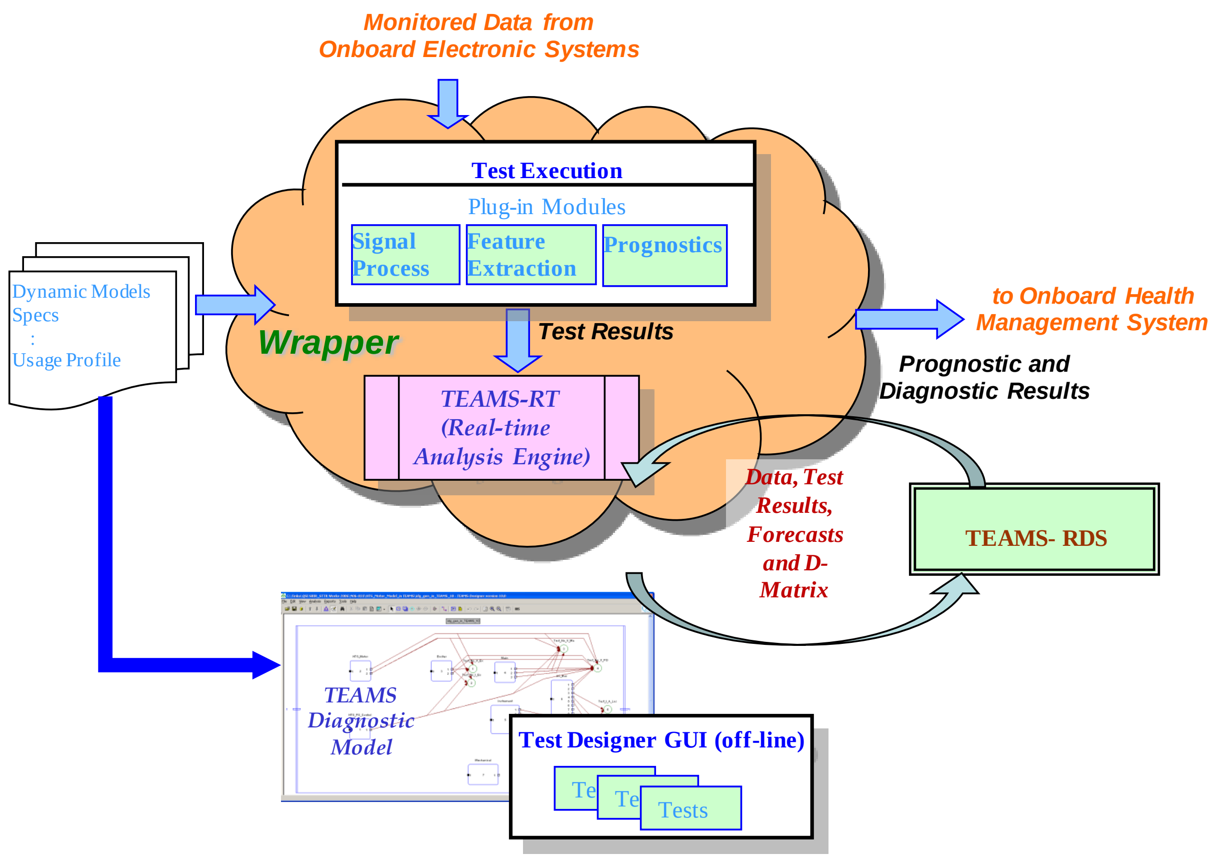

Figure 10: TEAMS-based Solution Framework for Automated Data-driven Prognostics

4.3.1 RTX Wrapper Development

QSI had chosen to realize the aforementioned functionalities and capabilities of the Automated Data-driven Prognostics Solution via a wrapper. The wrapper enables utilization of modeling and analytic capabilities of QSI's TEAMS modeling and analysis environment [5] [6]. TEAMS is a well established and matured platform that already has the essential elements of the aforementioned framework. Under this project, integration of these elements into an end-to-end prognostic solution along with its capability extension (in terms of algorithm hosting, execution, and data ingestion) is being performed. In the resulting solution, QSI's real-time reasoner engine, TEAMS-RT functions as a main decision-maker regarding diagnostics and prognostics. The regular TEAMS-RT works with binary test outcomes (pass/fail) only. For this project, via using the wrapper the capability of RT has been extended, such that raw data can be supplied as input (to the wrapper). This extended version of TEAMS-RT is named as TEAMS-RTX. The wrapper takes care of the issues related to data ingestion and decision exporting, on-demand execution of the tests, communication with the plug-in modules (e.g., optimal reconfiguration) and communication between TEAMS-RT and the Test Designer GUI. Models and inputs required for obtaining diagnostic and prognostic decision from the RTX can be directly ingested to the RTX Wrapper via TEAMS Designer and the APIs. However, there will be an added option to involve TEAMS-RDS [6] to control the flow of information to and from the RTX Wrapper. This option will be useful in the situations when the solution will be used for remote diagnostics and prognostics. The software architecture of the TEAMS-RTX wrapper is shown in **Figure 11**.

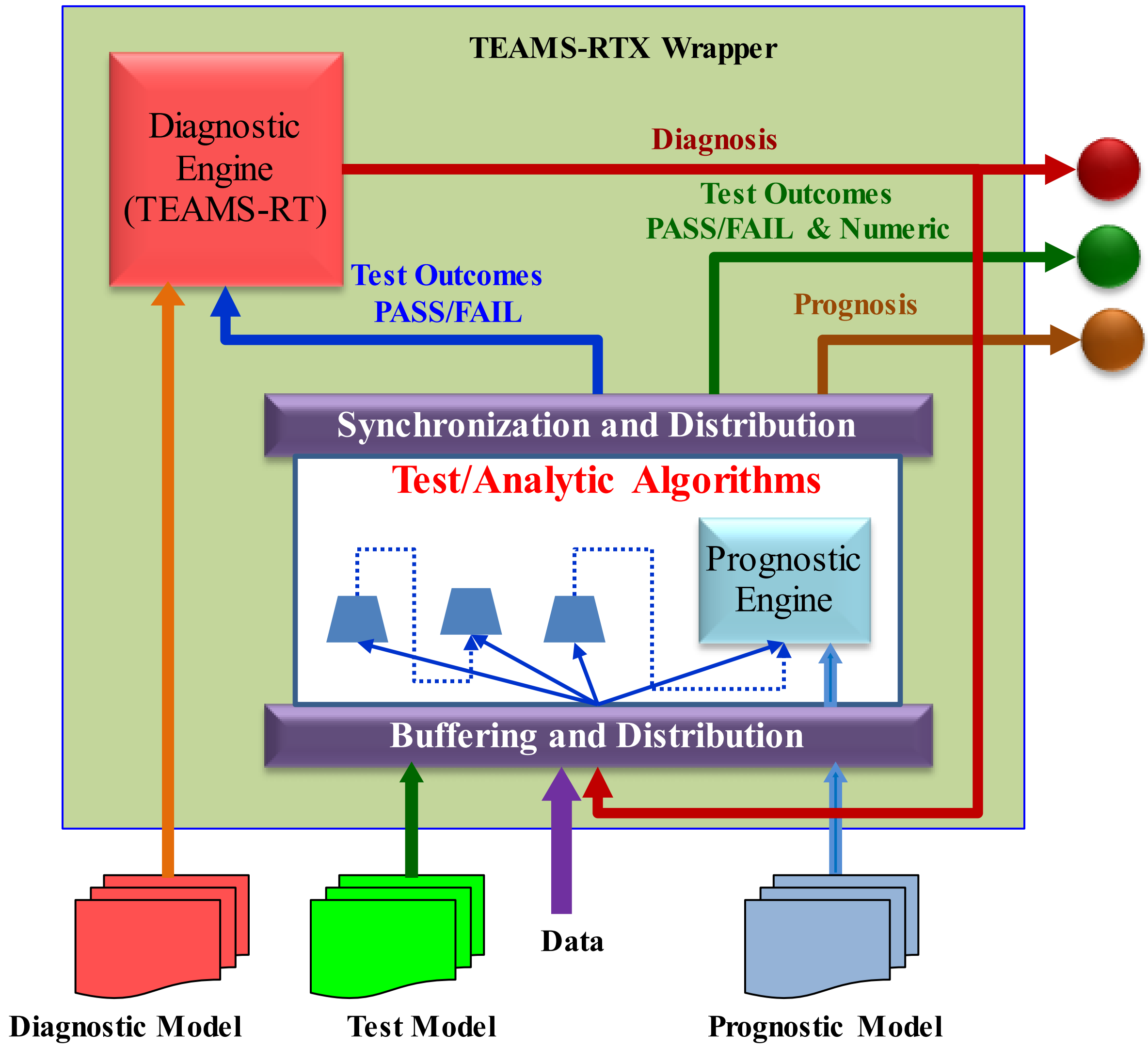

Figure 11: Software Architecture of the TEAMS-RTX Wrapper

RTX is a framework that allows application programs to use the functionality of the TEAMS-RT kernel as a diagnostic engine. In addition, the framework supports and integrates plug-ins. An RTX plug-in is a piece of executable code that performs a function (or functions) defined by its author. An existing plug-in gives an application the ability to perform the actions programmed into the plug-in without requiring the author of the application to write the code to implement them. Thus, existing plug-ins can save development time for the writer of an application. From another perspective, authoring a plug-in instead of adding functions directly to an application is a means of promoting code reuse in that the plug-in developed for the current application may be useful in a related future application, and can be used without any change whatsoever in the newer application. On operating systems that support such features, such as Windows and nearly all versions of UNIX, the plug-in is a dynamic-link library that is loaded into the address space of RTX (which also includes the RT kernel and the client application) as required and thereby becomes part of RTX. When such services are unavailable, such as on a simple imbedded system without an operating system, the required plug-ins must be linked with RTX, the RT kernel, and the main application to create the executable program. This document will focus primarily on using RTX on a Window operating system, although the overall operation and use of RTX differs little on other platforms and from the user's perspective, there is no change. Mostly, the differences are in terminology, such as the use of the term dynamic-link library (DLL) instead of a shared library or .so file, which would be the more common description used for Unix systems.

The wrapper architecture shown in Figure 11 employs a compartmentalized process for algorithm execution. An input buffer and distribution layer ensures appropriate data and/or information transmission to respective algorithms. The buffer takes care of data dimension, sampling interval, time synchronization between different sensor streams, etc issues as entailed by an algorithm. The output layer of the "algorithm execution compartment" performs the task of synchronization of analytic outcomes and distributing them to the appropriate diagnostic and/or prognostic engines. A diagnostic model describing how to utilize the analytic outcomes for real-time FDD, a prognostic model describing how to utilize the analytic outcomes for fault forecasting and remaining lifetime estimation, and a test model describing the data/information requirement and execution mechanism of the analytic algorithms control the function of the wrapper. These models can be simple text scripts, xml documents, or definition files in C. As these models are external to the actual wrapper, a user can define and revise them without touching the wrapper itself.

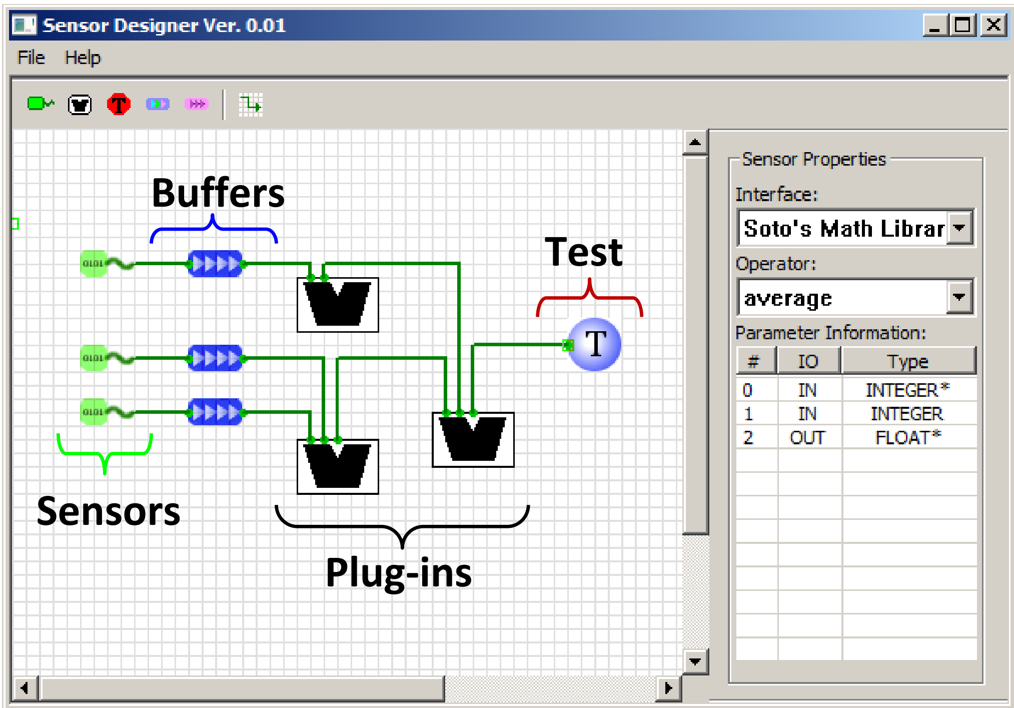

Figure 12: RTX Plugin Generator

As a part of this task, QSI has compiled and finalized a detailed user guide for the RTX wrapper. The user guide (a) explains how the RTX framework is initialized and provides an overview of its organization; (b) describes the initialization information; (c) provides information on how the TEAMS-RT diagnostic engine is used from within RTX, (d) discusses the rules and techniques on using existing plug-ins, and creating new plug-ins. QSI already completed some testing, modification and debugging of the code. Specifically, ingesting data from text files and sending diagnostic/prognostic outputs to a console application, and thereby testing the real-time performance of the solution was the major focus of the debugging and modification initiative. Several simple models with varying amount of data (ranging from 6 sensors each receiving a data vector of cardinality 4 at a sampling interval of 0.02sec, to 22 sensors each receiving a scalar at a sampling interval of 0.001sec) were used in these tests. The models employed diverse algorithms; the most complicated ones included GLRT, FFT, SVM, and some rudimentary algebraic operators. The task of testing, revising and debugging the codes continued during the May, 09 – Aug, 09 period

4.3.2 Test Plugin Generator Development

TEAMS-based analytic algorithms will function as "tests" for the TEAMS-based Automated Data-driven Diagnostics and Prognostics Solution. The usability of the solution largely depends on the ease of

i. Designing diagnostic and prognostic models
ii. Design of "Tests" and compiling them into a "Test Plugin"
iii. Interfacing the scheme with data sources (as a client application), and
iv. Interfacing the scheme with target output application

Among these, the issue of Designing Tests is of special interest, as these tests would mostly be designed by engineers who might not be familiar with the underlying codes. At present, the user needs to specify the interrelation of the algorithms in a test, buffer sizes, and synchronization information within the main application definition script. To allay this problem, QSI is developing a GUI based "Test Designer" and "Test Plug-in Generator" that will simplify the test designing process for RTX. A screen-shot of the "Test Designer"/ "Test Plug-in Generator" is shown in Figure 12.

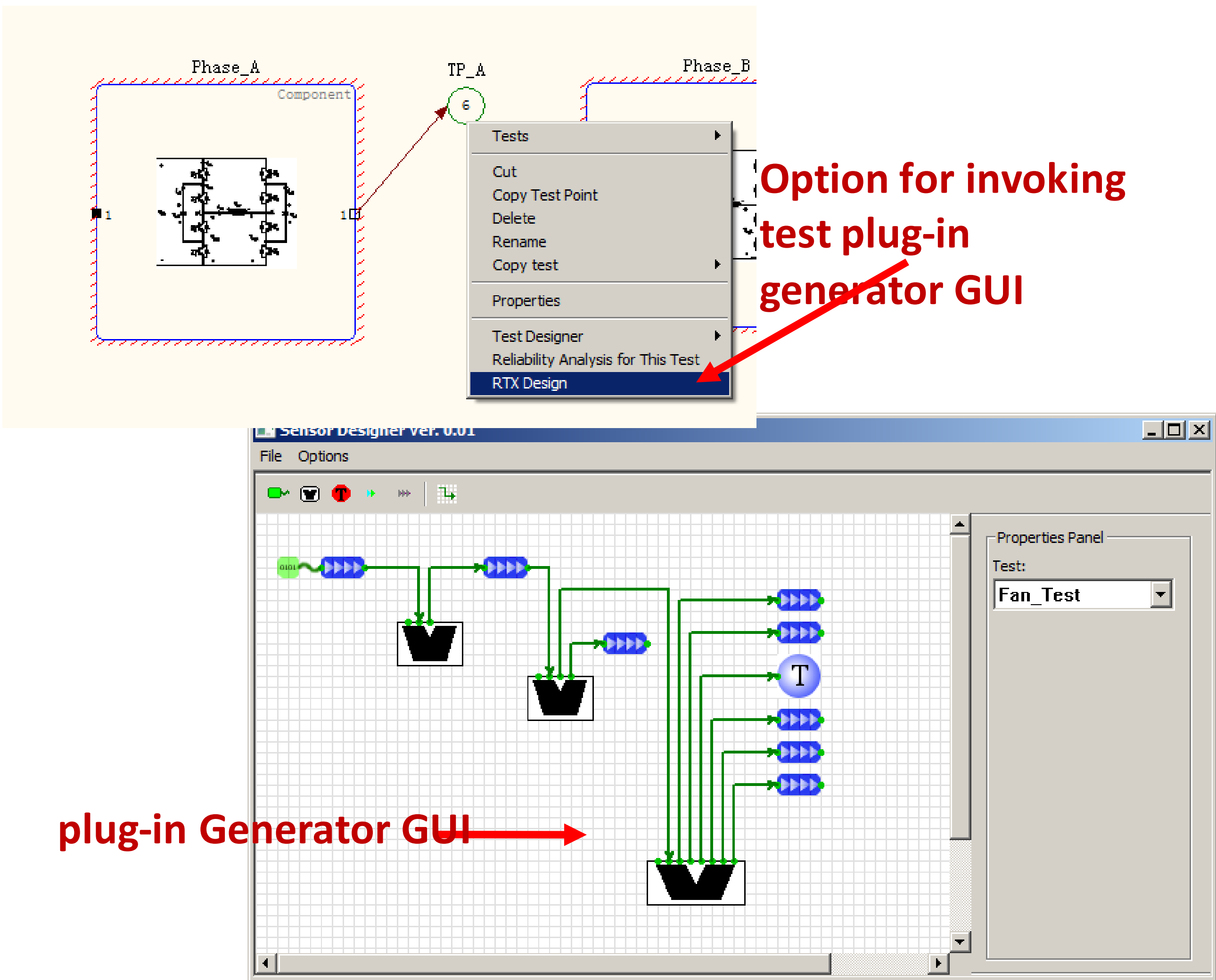

Figure 13: Invoking RTX Plugin Generator (Test Designer) GUI from TEAMS Designer

Introduction of this RTX Plugin Generator (Test Designer) GUI allows designing a test using multiple algorithms (each of the component algorithms needs to be in the pluggable dll form). It also facilitates the user to introduce buffers and synchronize the tests. The final version of the RTX Plugin Generator will allow saving tests in one model and reuse those partially and entirely in other models. Thereby, it will significantly reduce the burden of redesign and revalidation of tests.

For invoking the Test Designer GUI (which also functions as The Test Plugin Generator), an item, titled "RTX Design" has been added in the drop-down menu for a test point. By clicking on that item a Test Designer GUI screen can be opened. In the current version of the Test Designer, all test at a test point need to be designed on a single canvas. However, in some cases, number of tests at a test point can be considerably large (of the order of 10); in those cases, designing all tests on a single canvas becomes quite inconvenient. Reckoning this issue, QSI plans to facilitate individual Test Designer canvas for each test in a model. This plan will be realized during subsequent periods of performance.

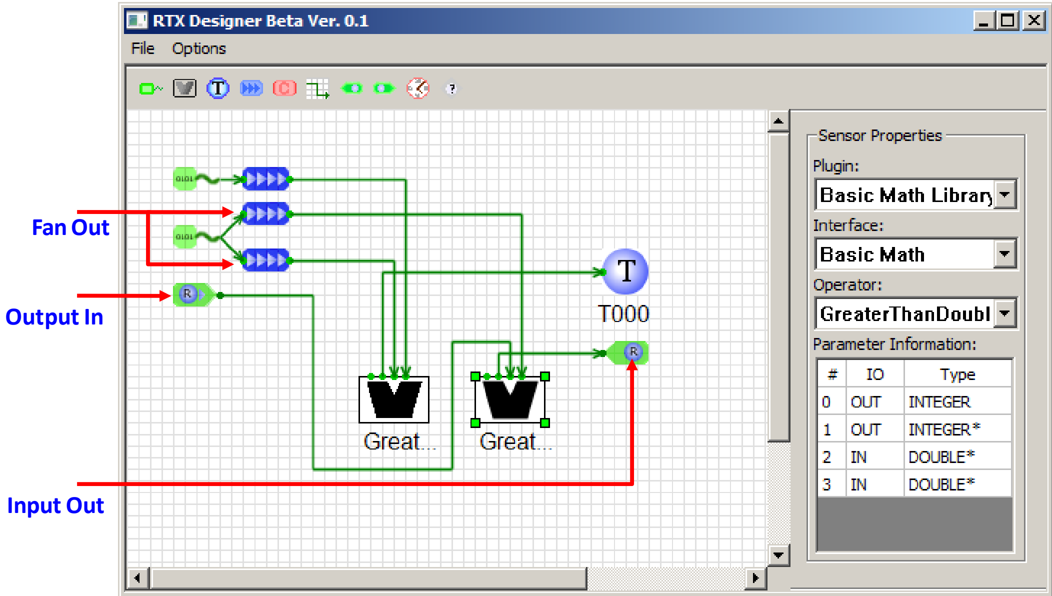

Figure 14: "Fan Out", "Input Out" and "Output In" Features in RTX Plugin Generator GUI

During the May 09 - Aug 09 period of performance, QSI introduces several new features to the RTX Plugin Generator (Test Designer) GUI. Foremost among the features are "Input Out" and "Output In" capabilities. The "Input Out" feature allows one (or more) processed/partially processed result(s) from a test plug-in design to another test residing under a different test point. The "Output In" feature allows bringing in processed results from a test into another test residing under a different test point. These capabilities eliminate the need of multiple executions of algorithms under a test that are also used in entirety under another test. QSI also worked towards allowing to copying a part of a test into another test. This capability will result substantial saving in time and effort for test design. To facilitate easy reuse of a sensor stream, "Fan Out" feature was introduced to the RTX Plugin Generator GUI. The above mentioned features are shown through an example design in Figure 14. Additionally, conditional operators are worked on during this period. The conditional operators will allow introduction of branching in a test design. Such a feature is essential for efficient realization of real-world test scenarios.

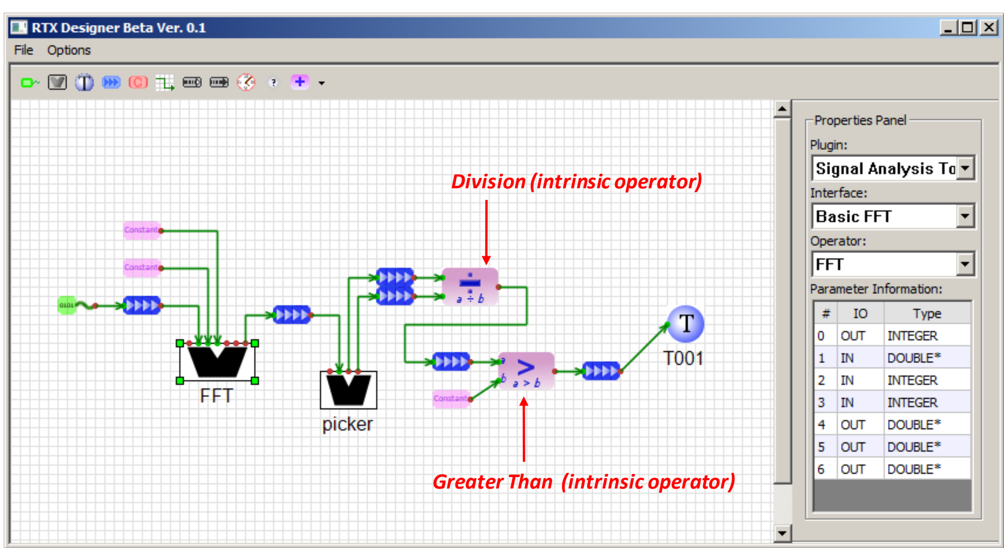

Figure 15: Intrinsic Operators In RTX Plugin Generator GUI

During the Aug 09 - Nov 09 period of performance, QSI introduced simple mathematical operators as intrinsic functions to the RTX Plugin Generator (Test Designer) GUI. The current set of intrinsic operators includes arithmetic (add, subtract, multiply, divide), relational (greater than, less than, etc), trigonometric, and some statistical (sum, mean, variance, etc) functions. The intrinsic operators make the test design process easier and reduce the processing time. Therefore, inclusion of these operators will ensure that a test is processed within the allowable time window for TEAMS-RT inference. In the Nov 09 -Feb 10 period, the issue of incorporating inherent buffers in Intrinsic Operators was resolved. This facilitates reduced burden on the test designer. A test using FFT and new intrinsic operators is shown in Figure 16.

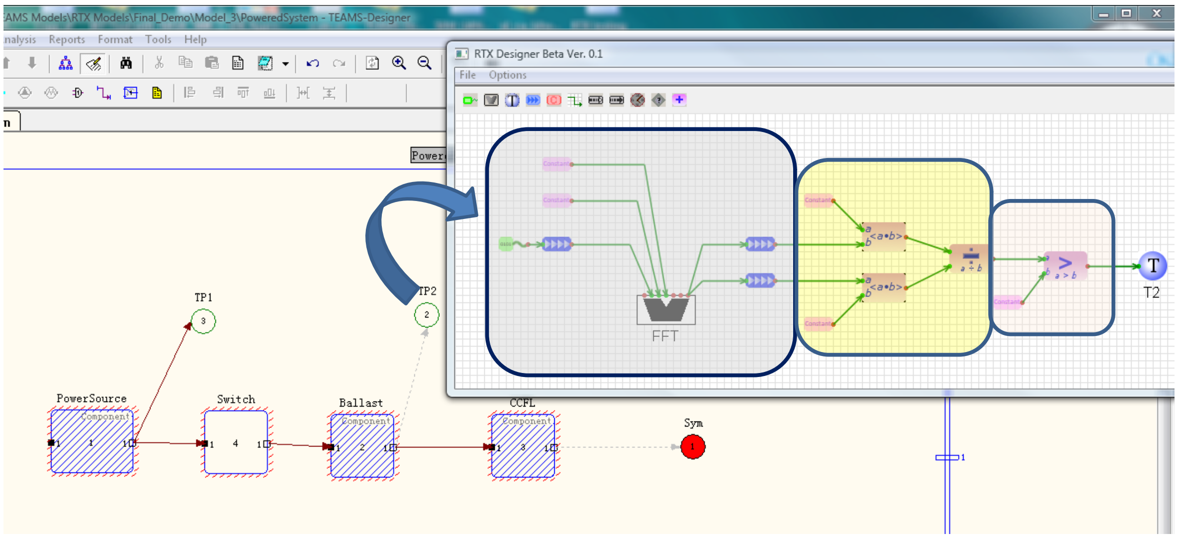

Figure 16: Updated Intrinsic Operators (with inherent buffer)

A major issue in real-time analysis is that the input (monitored) data stream might have, stale data, missing data, and “out of sequence” data. Such issues may severely affect the results of analysis. To allay such problem, during the Aug 09 - Nov 09 period of performance, QSI developed a set of techniques for restoring the sequence in a data stream, remove stale data, and substitute missing data. The out of sequence data issue is handled via two-stage sorting algorithm that reduces the sorting time significantly (i.e., allows to use complex operators that requires more processing time without violating the time allowance). Removal of stale data is an integrated functionality of the data sequencing algorithm; therefore no additional technique is required for this purpose. Substitution of missing/corrupt data entails two operations, (1) identification of missing/corrupt data, and (2) substitution with a value that follows the trend of the data stream. Standard outlier detection methods are not very suitable for this purpose; as such methods might identify an abrupt variation in data as a missing/corrupt sample. QSI experimented with a method that involves two stage outlier detection. In the first step, a sample is analyzed in relation with its adjacent samples to identify whether it is an outlier. If it is identified as an outlier, then it is analyzed in relation with a dictionary of outliers that comprises of samples belonging to previously missing/corrupt data (such a dictionary can be built a-priori using training data and enhancing it over time). This second stage outlier detection confirms the samples similarity/dissimilarity with other outliers. Once a missing/corrupt sample is identified, it can be substituted via interpolation, or averaging methods.

During the Mar 2010- May 2010 period of performance, RTX test designer GUI, sensors, buffers and the intrinsic operators for RTX were revamped. The updated versions of these elements facilitate more intuitive and easy to use designs. The intrinsic operators have been equipped with internal checkers for verifying that data passed to them conform to the data-type they are supposed work with. Internal checks are also performed to verify the cascadability of intrinsic operators during design-time. In addition, the process for exporting the RTX design was also made easier. Instead of exporting the design to a predefined location, now the user can choose the location through a browser window. RTX test designer canvas is now automatically populated with all the tests (and their associated outcomes) that are under the test point where the design is hosted.

4.4. Task-3: Data Generation using the RTCF Simulator

The target application platform of the prognostic solution developed from this project is onboard electronic systems of FCS platforms. Customization and modification of the solution for real-world FCS systems require experimental data from such systems. To fulfill this requirement, Northrop Grumman Corporation's (NGC) Integrated Systems Western Region is participating in this project as the data and expert knowledge provider. The simulated data will be generated using NGC's Real Time Component Framework simulator (RTCF SIM) using a prototype UAV Vehicle Management Computer (VMC), Virtual Line Replaceable Units (VLRUs) and some hardware prototypes. RTCF SIM functions as the core element for the task of data generation. The major advantage of using the RTCF simulator is that it replicates the complete UAV operation; i.e., it replicates the information, command, and decision flows as it would have been onboard a real UAV.

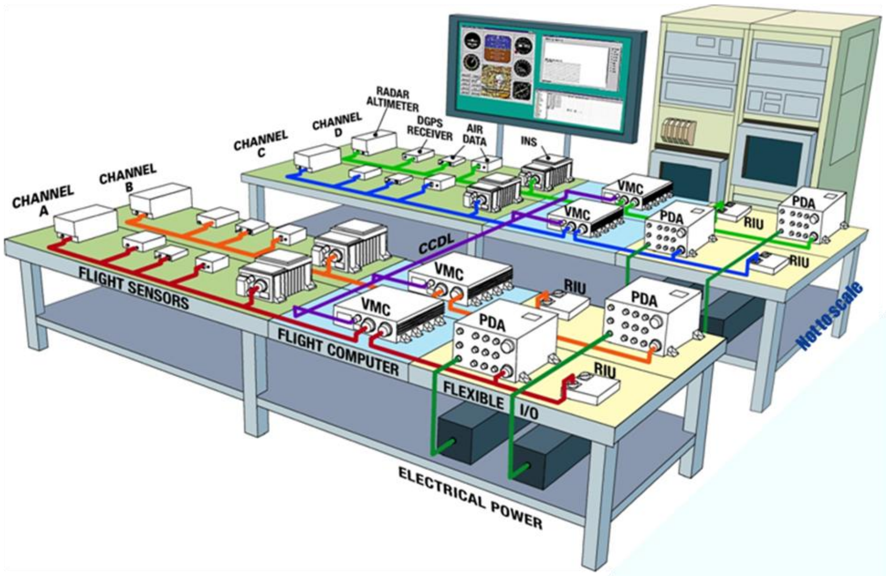

Figure 17: NGC Fire Scout VMS Hot Bench Development Environment

During the previous periods of performance NGC explored the strategies to provide simulated data for this project that suffice the requirements for data-driven prognostic algorithm testing and maturation. From the exploration it was found that the Fire Scout VMS Hot Bench Development Environment (that employs RTCF SIM) at NGC can comprehensively satisfy the data generation requirements (see **Figure 17**). The VMS Hot Bench Development Environment was constructed for supporting early technology and software demonstrations for existing and new UAV programs. Some salient features of this Environment that makes it suitable for this effort are

- Flexible interface architecture that supports subsystem integration for
  - Mission Management
  - Payload Sensors
  - Aircraft subsystems
- Baseline Flight Sensors
  - Radar Altimeter
  - DGPS Receiver
  - Air Data
  - INS

During this period of performance, NGC listed down the possible parameters (and their level of details) that might be obtained via simulation using RTCF. QSI and NGC plans to jointly examine the list and select the prospective parameters that will be of use in prognostic of the INS-GPS system.

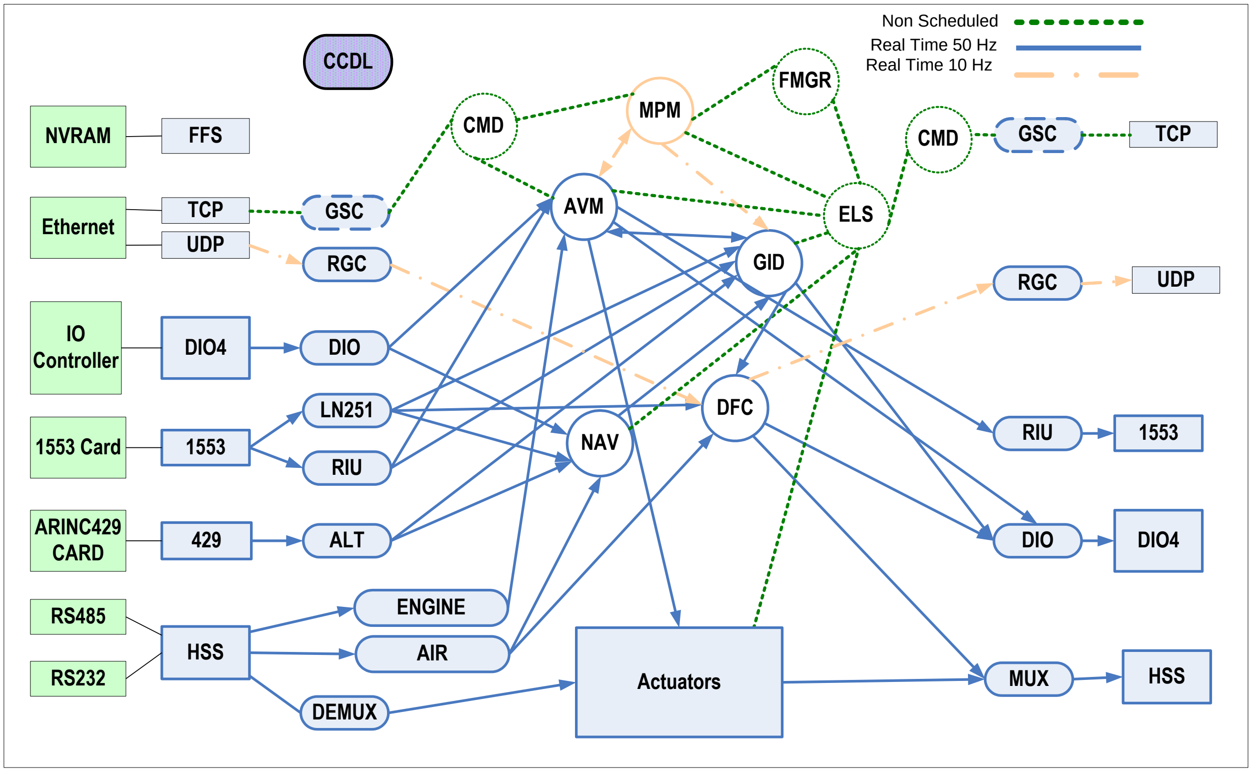

Figure 18: NGC's Generic Unmanned Air Vehicle Architecture (GUAVA)

NGC has opted for using Generic Unmanned Air Vehicle Architecture (GUAVA) software for this project. The GUAVA (Generic Unmanned Air Vehicle Architecture) software provides a simulation of an entire unmanned air vehicle, wherein the KN-4073 subsystem provides the identified subsystem for monitoring and analysis. An overview of the GUAVA architecture is given in Figure 18. NGC has configured an existing GUAVA model with a RTCF SIM of KN-4073 INS-GPS system and it will be tested against scenarios executing QSI's prognostic algorithms. The system allows modifications of nominal operations (in simulation) by injecting faults into the nominal operations of KN-4073 INS-GPS system to provide more realistic external indicators.

4.4.1 Parameter Selection for PHM Model Development of KN-4073

During the Mar 09 – May 09 time period, QSI and NGC jointly investigated the parameters that could be used in condition monitoring and health forecasting of the KN-4073 INS-GPS system. Since, the RTCF will perform simulation using a mathematical model of the KN-4073 INS-GPS, we decided on using the Kalman filter parameters as the inputs for the condition monitoring, and health and/or performance forecasting scheme. A list of all available Kalman filter parameters is shown in Table 2.

Table 2: Kalman Filter Parameters for the KN-4073 INS-GPS

Parameter Name	Parameter ID	Unit	Data Type
System Timer	1-8	sec	dp fl pt
GPS Time	9-16	sec	dp fl pt
Sin alpha sigma (IFA)	17-20	rad	sp fl pt
Cos alpha sigma (IFA)	21-24	rad	sp fl pt
Spare	25-32	N/A	N/A

X angular position 1-sigma	33-36	rad	sp fl pt
Y angular position 1-sigma	37-40	rad	sp fl pt
Altitude 1-sigma	41-44	m	sp fl pt
X velocity 1-sigma	45-48	m/sec	sp fl pt
Y velocity 1-sigma	49-52	m/sec	sp fl pt
Z velocity 1-sigma	53-56	m/sec	sp fl pt
X tilt 1-sigma	57-60	rad	sp fl pt
Y tilt 1-sigma	61-64	rad	sp fl pt
Platform azimuth 1-sigma	65-68	rad	sp fl pt
X gyro bias 1-sigma	69-72	rad/sec	sp fl pt
Y gyro bias 1-sigma	73-76	rad/sec	sp fl pt
Z gyro bias 1-sigma	77-80	rad/sec	sp fl pt
Gravity Anomaly	81-84	m/sec2	sp fl pt
Spare	85-92	N/A	N/A
X gyro scale factor 1-sigma	93-96	ppm	sp fl pt
Y gyro scale factor 1-sigma	97-100	ppm	sp fl pt
Z gyro scale factor 1-sigma	101-104	ppm	sp fl pt
X accel bias 1-sigma	105-108	m/sec2	sp fl pt
Y accel bias 1-sigma	109-112	m/sec2	sp fl pt
Z accel bias 1-sigma	113-116	m/sec2	sp fl pt
X correlated accel bias 1-sigma	117-120	m/sec2	sp fl pt
Y correlated accel bias 1-sigma	121-124	m/sec2	sp fl pt
Z correlated accel bias 1-sigma	125-128	m/sec2	sp fl pt
X accel scale factor 1-sigma	129-132	ppm	sp fl pt
Y accel scale factor 1-sigma	133-136	ppm	sp fl pt
Z accel scale factor 1-sigma	137-140	ppm	sp fl pt
GPS clock phase 1-sigma	141-144	met	sp fl pt
GPS clock frequency 1-sigma	145-148	m/sec	sp fl pt
GPS correlated clock frequency 1-sigma	149-152	m/sec	sp fl pt
GPS clock X g-sensitivity 1-sigma	153-156	m/sec/g	sp fl pt
GPS clock Y g-sensitivity 1-sigma	157-160	m/sec/g	sp fl pt
GPS clock Z g-sensitivity 1-sigma	161-164	m/sec/g	sp fl pt
GPS correlated pseudorange bias 1-s (ch 1)	165-168	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 2)	169-172	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 3)	173-176	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 4)	177-180	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 5)	181-184	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 6)	185-188	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 7)	189-192	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 8)	193-196	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 9)	197-200	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 10)	201-204	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 11)	205-208	m	sp fl pt
GPS correlated pseudorange bias 1-s (ch 12)	209-212	m	sp fl pt
Baro altimeter bias 1-sigma	213-216	m	sp fl pt
Baro altimeter scale factor 1-sigma	217-220	ppm	sp fl pt

NGC will investigate the usefulness of these parameters in condition monitoring, and health and/or performance forecasting using QSI-UMD's algorithms. Down selection of the parameters and regarding their contributions in health/performance forecasting will be achieved through this study for usefulness.

4.4.2 Experiment in GUAVA Environment

As a precursor to generating data using the KN-4073 on the RTCF platform, during the May, 09 – Aug, 09 period of performance, QSI and NGC performed some experiments using NGC's GUAVA environment and a flight simulator model (software) of Fire Scout UAV. The experiments involved maneuvering the UAV along a predefined course; obtain position, velocity, altitude, and acceleration related observations; and injecting faults/disturbances in the observed parameters. The goal of the experiment was to verify that the developed HM techniques are able to differentiate between nominal and faulty data. The faults/disturbances were injected via a manual post-data collection application. The application is able to inject several types of disturbances, including spikes, step, ramp, square wave, triangular, sinusoidal, and hyperbolic functions of different magnitude and duration. Disturbance might comprise of a single function or a series of functions; those functions can be of identical or different types, and they can either be time separated or overlapped.

Data was generated for circular flight paths with varying altitudes. It has been planned to use some of the nominal data as training patterns, and to create faulty patterns by injecting faults onto a mix of training and different nominal (those are not part of the training patterns) patterns. This would enable the detection performance of the HM techniques. In this simulation scenario, faults can only be injected in the parametric level; therefore, only detection of faults and identification of the faulty parameter can be performed; but, the root cause behind the faults cannot be identified. As an extended part of the experiment, the QSI-NGC team plans to feed the faulty data (data with injected faults) back into the simulator and observe the subsequent changes in the monitored parameters. Performing such feedback continuously is somewhat analogous to fault progression over time. Therefore, this data provides an opportunity to verify the performance of the STSA-based prognostic scheme. The scheme will use a sequence of observations as training patterns, and will forecast the parameters based on those observations. Forecasting schemes will be verified for both observed parameter (viz., when the forecasted parameter is a member of the set of input parameters used in its forecasting), and unobserved parameter (viz., when the forecasted parameter is not a member of the set of input parameters used in its forecasting).

Upon verification of the condition monitoring and prognostics techniques on the data generated via the GUAVA environment using the UAV simulator, the QSI-NGC team will either generate data using the RTCF testbed or the Fire Scout HIL (hardware in the loop) simulator. Selection of the next stage data source will depend on associated costs and types of experiments that can be performed on these respective platforms.

4.4.3 Generating Data through HIL Experiment on KN-4073 INS-GPS

In early June of 2010, QSI and NGC performed HIL simulation experiments using KN-4073 INS/GPS unit. The experiments were conducted at Kearfott Corporation's[‡] Little Falls, NJ facility with the help of their engineering personnel. The data collected during this experiment served the purpose of providing the "feel" of the real-world data on which the techniques developed from this project will actually be working on. The data also provided an opportunity to perform preliminary test of the condition monitoring algorithms developed through this effort. A brief discussion on the simulation scenarios and the collected data follows.

4.4.3.1 HIL Experiments with KN-4073 INS-GPS

The major goal of the experiments was to develop parametric models for the INS-GPS unit at its nominal and degraded conditions. The degradations were simulated via

- Introducing bias in Accelerometers, Gyroscopes

[‡] The manufactruere of KN-4073 INS/GPS

- Adding impedance in signal path
- Restricting DC power to the pre-amplifier

The simulation scenarios encompassed degradations that affect the "electronic and electro-mechanical" and "purely electronic" components in the system. Reduction in signal strength simulated degradation in antenna and associated electronic components, reduction in power imitated problem/degradation in the pre-amplifier of the GPS assembly. Bias in accelerometers and gyroscopes imitated degradation in those subsystems, respectively. A wide range of impedance variation (ranging from 100Ω - 100KΩ) and DC voltage variation (ranging from 5.0V-0.6V) were used in the simulation runs. Since accelerometer and gyroscope bias related observations need to be recorded over a longer time span (10-20min), only a few simulation runs were performed involving accelerometer and gyroscope degradation. The INS/GPS has several modes of operation; among those, Hybrid Navigation mode was chosen for the simulation experiment. The reason behind the choice was that in this mode both the INS and GPS subsystems are used for navigational purpose.

The KN-4073 has above 100 parameters that are monitored during simulation runs. Most of the parameters are sampled at 1Hz (e.g., ISA Raw Data Elements and EGR Line of Sight Data[§]); however there are some parameters that are sampled at 50Hz and 100Hz (Navigation IMU Data and Autopilot IMU Data[**]) as well. Although not all these parameters are not of value for condition assessment, diagnostics, and prognostics, they were collected to verify the efficacy of the data ingestion scheme. Exploring the possibility of apparently unimportant observations (e.g., EGR CV Status Message A, EGR CV Status Message B, etc) in prognostics was another goal for collecting the whole spectrum of the data.

4.4.3.2 Conversion of Data

The raw data obtained via KN-4073 simulation was encoded. Most parameters were encoded according to Kearfott's custom coding schemes, while others were encoded according to some standard formats, such as IEEE 754 Format Single/Double Precision, DEC Format Single/Double Precision [45], Collin's Adaptive Signal Processing (CAPS)[††] etc. Most parameters were expressed through individual messages, but some were bundled within a single message. The lengths of these messages were different as well (ranging from 1bit-64bit). The data was received in 11 bit packets consisting of a start bit, 8bits of data, a parity bit, and a stop bit.

To make the data usable with the diagnostic and prognostic algorithms it was required to convert them into sequence of real numbers. Under this subtask, QSI developed converters for automatically extracting messages from the data stream and then transforming them into time-series of real-numbers representing the values of system parameters. During early June of 2010, the data was converted to .csv and subsequently to .mat format such that analytic techniques developed in MATLAB/SIMULINK along with those already implemented in TEAMS-RTX can all be customized for the KN-4073 INS/GPS.

[§] ISA → Inertial Sensor Assembly; EGR → Embedded GPS Receiver
[**] IMU → Inertial Measurement Unit
[††] A proprietary data format of Kearfott Corporation

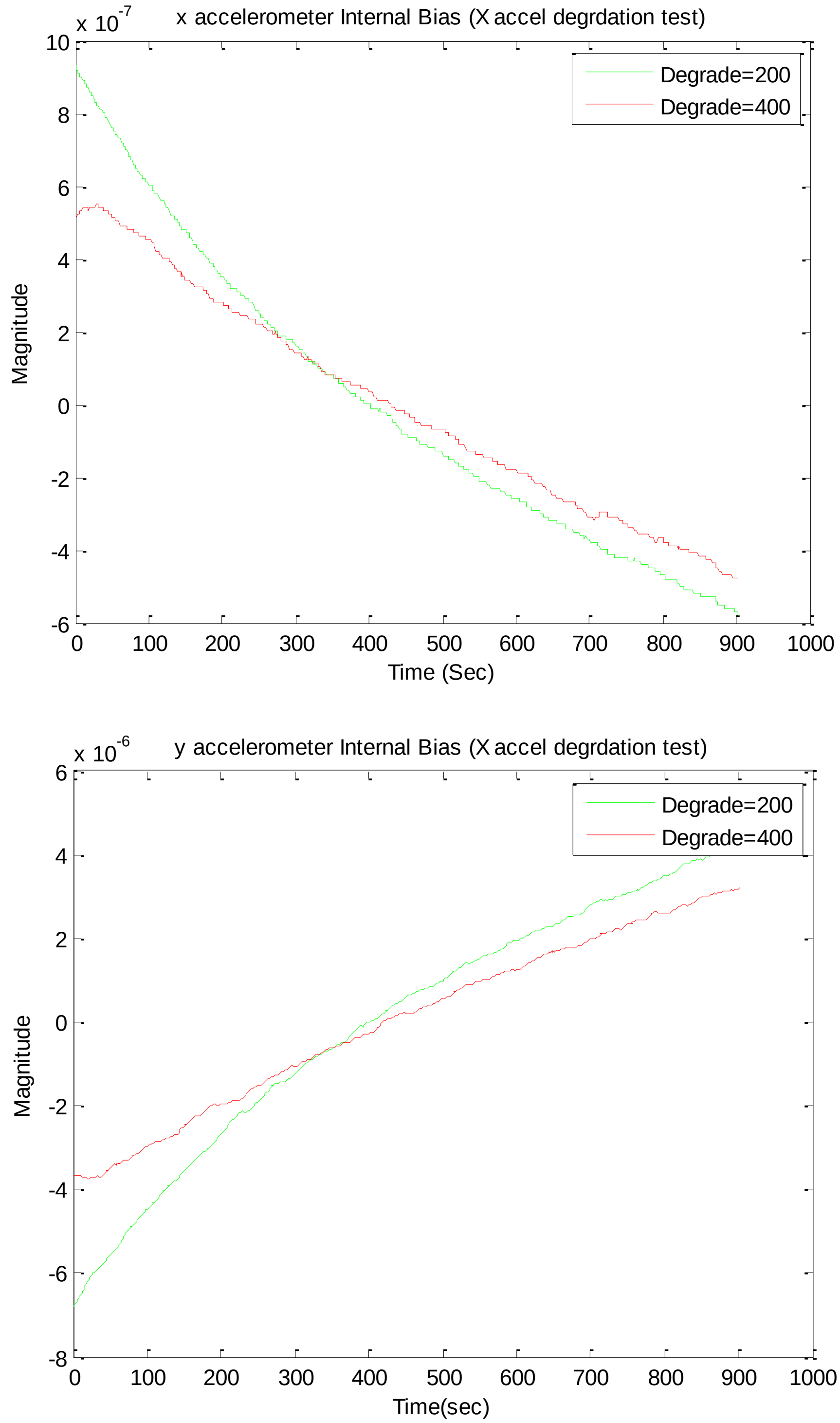

Figure 19: Accelerometer Internal Bias Variation with Degradation in X Accelerometer

These data were plotted and analyzed for identifying the trend of variation at different degradation settings. Example cases for biases at X and Y accelerometers for degradation in X accelerometer are shown in Figure 19. The plots also provided the preliminary information about usability of the parameters in diagnostic and prognostic analysis. Such information helped us in developing a more elaborate simulation plan (to be performed under Task-5 in the final quarter of this project).

4.5. Task-4: Testing Validation and Refinement

Data from the KN-4073 simulation simulator will function as the basis of customization of the data-driven prognostic and condition monitoring scheme that was developed during the Phase-I of this project. Additional algorithms and techniques that might deem necessary for fault and degradation prognostics in the target platform will also use the simulation data from the RTCF or the Fire Scout HIL simulator. A substantial part of the Phase-II effort will be dedicated towards validating and refining the customized prognostic solution. The QSI team will employ a scheme comprising of both manual and automated (or semi-automated) to perform this task (see Figure 20). In customizing a generic diagnostic/prognostic solution for a specific system the major errors and non-conformances arises from differences in the nature of monitored data (e.g., data rate, resolution, latency, etc), specific system events (e.g., mode switching, scheduled interrupts, etc), and not accounted for operational and environmental conditions. Majority of the errors and non-conformances emanating from the first two sources can be fixed via single-shot re-designing of the prognostic approach and algorithm modifications. However, continuous modification is needed to adapt the solution for a broad range of operational and environmental conditions.

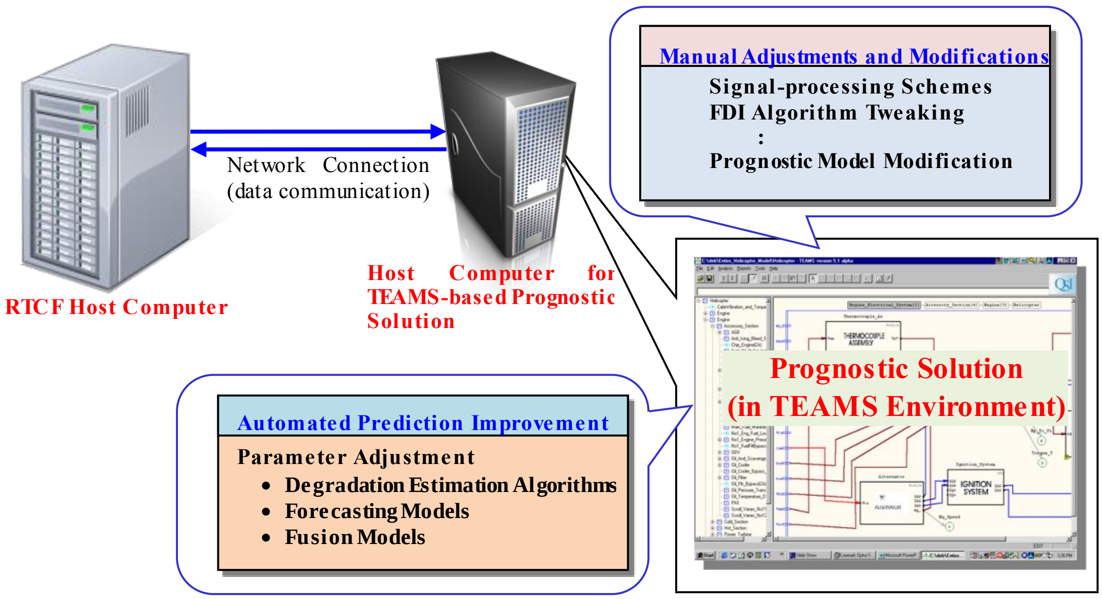

Figure 20: Modification and Adaptation of the Prognostic Scheme

Upon obtaining simulated data from the RTCF, the QSI team will initially evaluate the functionality of the data preprocessing and feature extraction schemes. It is common to experience out of sequence data, varying data rate, stale and missing data during validation runs. Depending on the nature and severity of such events QSI and UMD will modify the aforementioned parts of the prognostic solution; the modification might also involve re-structuring the prognostic solution. A target objective of the Phase-II effort is to deploy the prognostic solution onto a test-bench. We plan to fulfill this objective by embedding the prognostic scheme in QSI's TEAMS modeling and analysis environment and then deploying it to the RTCF environment. Since the archived simulated data might not encompass monitored data having of all possible characteristics (in terms of values, combinations and anomalies) some modification to the preprocessing and feature extraction scheme might become essential after deployment of the TEAMS-based prognostic solution to the RTCF environment. Majority of such post-deployment modification will be performed via introducing patches and plug-ins so as to minimize the requirement of re-deployment.

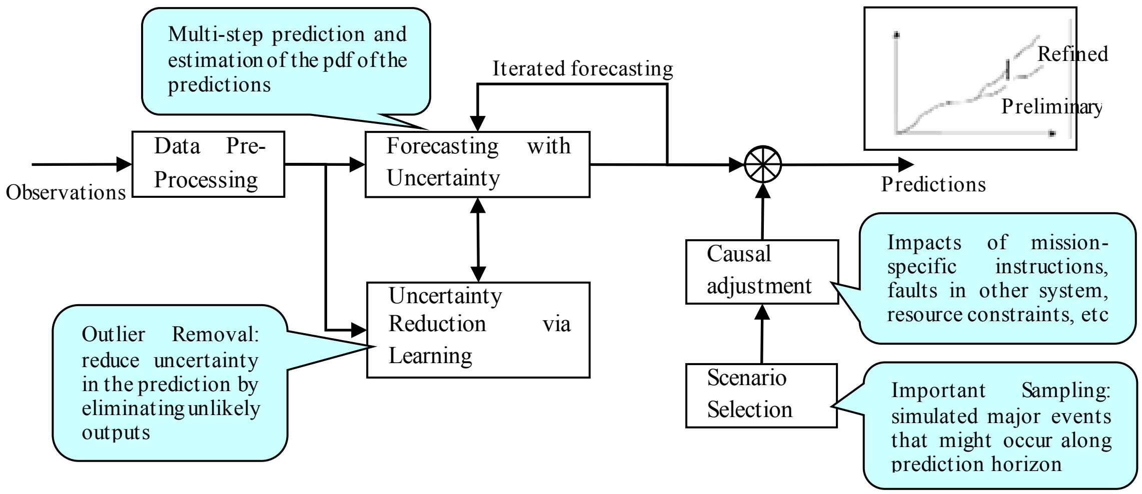

Figure 21: Improvement in Prediction via Reinforcement Learning

An iterative approach will be incorporated for improvement of the prognostic and degradation/failure source identification capabilities of the developed solution. A part of the data generated from the RTCF testbed will be used for initial validation. Upon reaching an acceptable level of accuracy, validation on scenarios generated via Monte-Carlo simulations will be performed to enhance the adaptability of the developed techniques to possible behavioral deviations of the ultimate target system from the prototype which would be used for prognostic solution development. The accuracy improvement initiative will continue in the post deployment Phase of the prognostic solution. We expect to experience operational scenarios, usage and environmental conditions beyond those which have been used to customize the prognostic solution during the post deployment runs. An automated reinforced learning-based approach will be embedded into the solution to continuously improve the prognostic performance upon experiencing novel situations where the prognostic accuracy happens to be unsatisfactory.

Confidence bounds associated with the forecasting results will be used as the primary measure for performance evaluation of the parameter prediction scheme [23]. Data-driven confidence prediction neural network architecture will be used (see Figure 21) that targets intended to 'shrink' the uncertainty bounds. To aid the accuracy enhancement of prognostic techniques, specifically, for the parameter forecasting part a semi-automated process will be followed. This process will be based on the principle of reinforced learning (see Figure 21) [24]. For NN based forecasting technique the scheme can be incorporated just in the form shown in Figure 21. The system will be implemented for time-series based forecasting methods by modifying the algorithmic parameter update rules.

During the May 09 – Aug 09 period of performance, a test environment in NGC was set up that resembles the one shown in Figure 20. A Windows-based server hosting the GUAVA environment and UAV simulator mimicked the RTCF host computer. Another Windows-based PC running the STSA-based analytic tools assumed the role of TEAMS-based prognostic application host. The setup was tested offline to verify their functionality. It involved fault injection into data obtained via experiment on the Fire Scout simulator (see Section 4.4.2 for details) and analyzing that using the STSA-based algorithms. In the upcoming period of performance, a more realistic testing of the proposed solution will be performed by using a TEAMS-based prototype (that will subsume the present STSA-based analytic tools) of the proposed solution.

During the Aug 09 – Nov 09 period of performance, QSI tested their fault detection, classification, and forecasting algorithms embedded in the TEAMS-RTX on data from Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) [25] data available from NASA's DASHLink website. Additionally, the forecasting algorithms were verified against data simulated from a simple amplifier circuit built using MATLAB Electronic Systems Toolbox. We also verified the functionality of TEAMS-RTX based solution with tests that utilize two or more advanced algorithms and synchronization between the algorithms is a must. Issues involving continuous feed of data through csv files (where a csv file is appended with new rows in real-time) were also investigated during this period of performance. Several modifications in the data ingesting process were incorporated through this investigation. QSI will continue the V&V work over the remaining period of this STTR. In the upcoming period, we expect to verify our techniques using data from a prototype onboard electronic system.

4.5.1 Assignment of Confidence to Prognostic Outcomes

During the Nov 09- Feb period of performance, the QSI-UMD team looked on the prospect of developing a complementary forecasting scheme that can be used to verify the accuracy and consistency of the results from STSA. In essence, the complementary schemes results are compared with the results from STSA, and when they conform, the result from STSA is labelled with a "high confidence level" and vice-versa. Such confidence levels are essential in identification of failure sources when the tests are imperfect [28]. The QSI team selected Self Organizing Map (SOM) as the complementary prognostic scheme. A description of SOM and the performed work is given in the following subsection.

4.5.1.1 Self Organizing Map (SOM)

The Self Organizing Map (SOM), also known as the Kohonen map, is a neural network algorithm for the visualization of high-dimensional data. It converts complex, non-linear statistical relationships between high-dimensional data into simple geometric relationships on a low-dimensional display. The main advantage of using SOM is its orderliness of the input-output mapping which can be utilized in many tasks: reduction in the amount of training data, speeding up learning, nonlinear interpolation and extrapolation, generalization, and effective compression of information for its transmission [29][30]. The SOM is an unsupervised technique that can be regarded as clustering, visualization, and abstraction. The technique consists of taking a data set that can be high dimensional and organizes it in a visual map which can convert complex and non-linear statistical relationships between high-dimensional data into simple geometric relationships on a low-dimensional display. SOM can also show the similarities within a data set. The SOM method has been widely used in fault detection and diagnosis; however, very little research has been done on using it for prognostics. We propose the use of a Self Organizing Map (SOM) to monitor the fault progression in electronic systems and will address the use of Back Propagation (BP) Neural Networks (NN) to predict their remaining useful life.

4.5.1.2 Feasibility of using SOM in Prognostics of Electronic Systems

The motivation for using SOM for PHM of electronic systems is the fact that these systems have a multitude of available measurable parameters such as temperature, voltage, resistance, and many others. Also, there may exist some correlations between these parameters. By using the SOM, the dimensions of the data matrix will be reduced, and similarity will be shown; this will help determining the correlations by producing geometric relationships or clustering. This will help to identify changes in the system, and to identify the important parameters to monitor. Furthermore, SOM can also be used for fault identification by creating a healthy map from normal operating data, comparing the healthy map with the test data, and finally creating simulated maps for specific faults. Figure 22 shows a scatter plot for data collected from computers and an organized map as an output from the SOM package developed at CALCE.

SOM has found many applications in science and engineering and other fields. It caught the attention of many researchers; the following list is by no means extensive, but includes some of the areas where SOMs have been successfully used [29]: visualization of machine states, fault identification where the abnormal state can be compared to the normal state of the system, process analysis and monitoring, computer vision, texture analysis classification, speech recognition, robotics, robot navigation, and telecommunication. A number of researchers used the SOM algorithm for health management. Often the work served the detection or the diagnosis functions of health management, and a few research addressed the prognostic or the prediction part of PHM. Gebrael et al. (2004) assessed the remaining useful life of bearings from vibration-based degradation signals using a neural network approach. The authors used the Weight Application for Failure Time (WAFT), Weight Application for Exponential Parameters (WAEP), and Weight Application to Exponential Parameters-parameter updating (WAEP-UP). Their result show that the remaining useful life for bearings was found that 64% of predictions are within 10% of actual bearing life, while 92% of predictions are within 20% of the actual life [31]. Huang et al. (2007) built on the work performed in [31] and used the Mean Quantization Error (MQE) that is output from the SOM algorithm along with a neural network algorithm and used the WAFT developed in [31] to estimate the remaining useful life of bearings. The authors showed that the method developed is superior to the L10 method currently being sued [32]. Chai et al. (2005) used artificial neural network for detecting faults and determining their locations in communication optical fibers. They built a topological model of the network and used their ANN algorithm successfully for fault detection and isolation [33]. Cottrell et al. (2009) used self organizing maps for fault prediction in aircraft engines. The authors designed a procedure to visualize successive data measured on an aircraft engine, and use self organizing maps to project multi-dimensional data and track the measurements over time. They analyze the trajectories, and observe deviation from normal behavior which may lead to the conclusion of anticipating a fault [34]. A body of literature exists about the use of self organizing maps for fault detection and diagnosis; some examples can be found in [35][36][37][38][39].

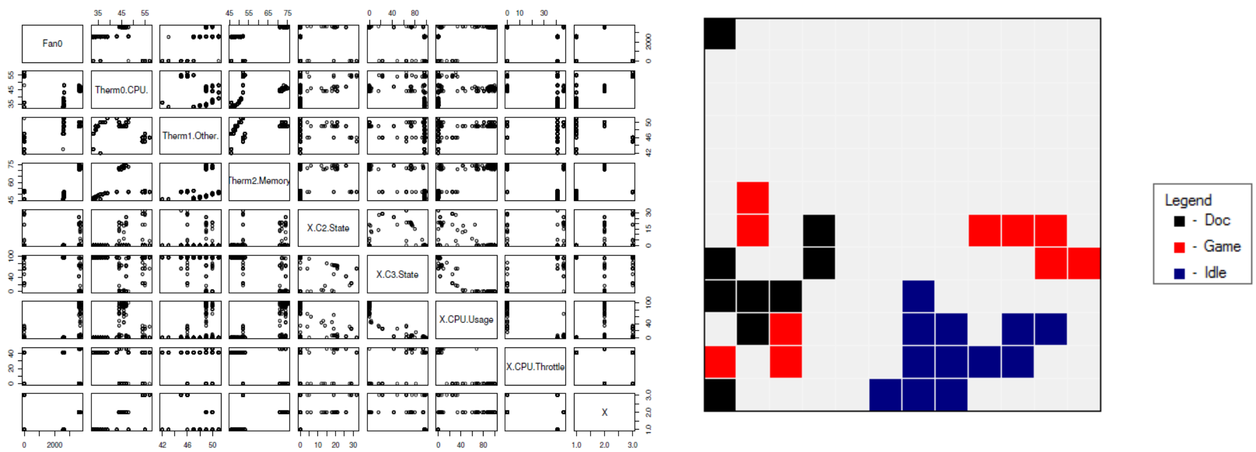

Figure 22- a) Scatter plot; b) map from software developed at CALCE The Kohonen Algorithm

The training of the algorithm occurs in several steps: data is normalized and a map size is determined. Then each node's weight is randomly initialized. Then a vector is chosen at random from the set of training data and presented to the lattice. Every node is examined to calculate which one's weights are most like the input vector. The winning node or neuron is commonly known as the Best Matching Unit (BMU). The radius of the neighborhood of the BMU is then calculated. This value starts large and is set up to the radius of the lattice and diminishes with increments of time steps. Any nodes found within this radius are deemed to be inside the BMU's neighborhood. Each neighboring node's weights are adjusted to make them more like the input vector. The steps are then iterated. The overall description of the algorithm can be seen in Figure 23. It consists of 6 major steps: normalizing data, determining the map size, initializing the weights vectors, find the winning neuron, find the neighborhood, update the neurons, and the process is iterated.

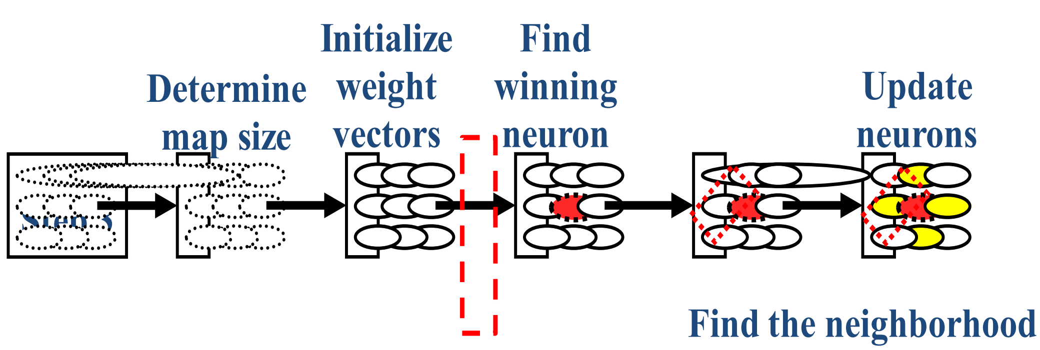

Figure 23- Steps in the SOM algorithm

After normalizing the data and choosing the map size, the next step involves assigning random weights to each of the neuron in the lattice. Then an input vector is chosen from the data set, and the Euclidian distance between the input vector and each neuron is calculated. The distance calculation for all the neurons in the lattice is repeated, and the winning neuron is chosen to be the one with the minimum distance with the input vector. Next, the wining neuron selects its neighbors, and determines which neurons will be updated; typically they are the ones from its neighborhood. The neighborhood definition is from a predefined neighborhood function. Then the weight vectors are updated according to a predefined function too:

$$W_i(t+1) = W_i(t) + h_{ci}(t)[x(t) - W_i(t)] \qquad (e.4)$$

Note that the weights are updated for neurons inside the neighborhood function. Then the steps are repeated for each vector from the input data set.

4.5.1.3 Modification and Proposed Method

SOM's have primarily been used for fault detection and diagnosis. For these two particular functions of PHM, we develop a map of the healthy state of the data which can be compared to test data; this will provide process monitoring, and the deviation from the healthy state can be identified as an anomaly. Another way of using the SOM for these functions of PHM is to track points on the map and check whether the progression with time goes to an anomalous value. This can be seen in Figure 24. Also, fault detection can be assessed by studying the deviation of the quantization error from the normal feature space.

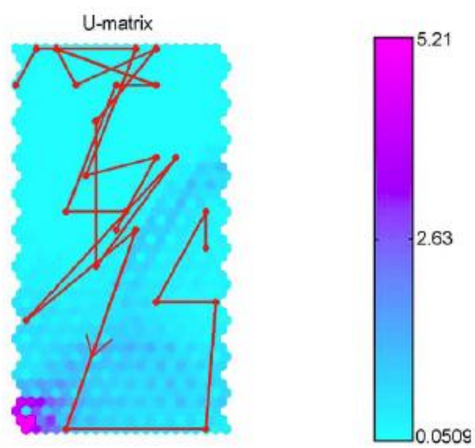

Figure 24- Tracking points on the map

The SOM algorithm's function is extended and is used to predict the remaining useful life where it is used with a back-propagation neural network. At first, the SOM is trained with normal operation data. Then the feature is compared with the weight vectors of all map units, and if the smallest difference exceeds a threshold, the system is considered to be fault situation. The degradation assessment can be determined by the calculation of the minimum quantization error (MQE) of the new measurement data to a SOM trained using normal operation data sets. From a degradation monitoring point of view, the distance between the BMU and the input data actually indicates how far the input data deviate from the region of normal operation. Thus the magnitude of degradation can be quantized and visualized by following the prevalent pattern of the MQE. Qiu et al. (2003) introduced the MQE index and applied it successfully as a degradation index [39]. A further enhancement of the technique is the use of back-propagation algorithm along with the Weight Application of Failure Time method on the mean quantization error that is output from the Kohonen map in order to calculate the remaining useful life [32][39]. Thus the use of SOM will serve a third function of health management besides fault detection and diagnosis: prognostics.

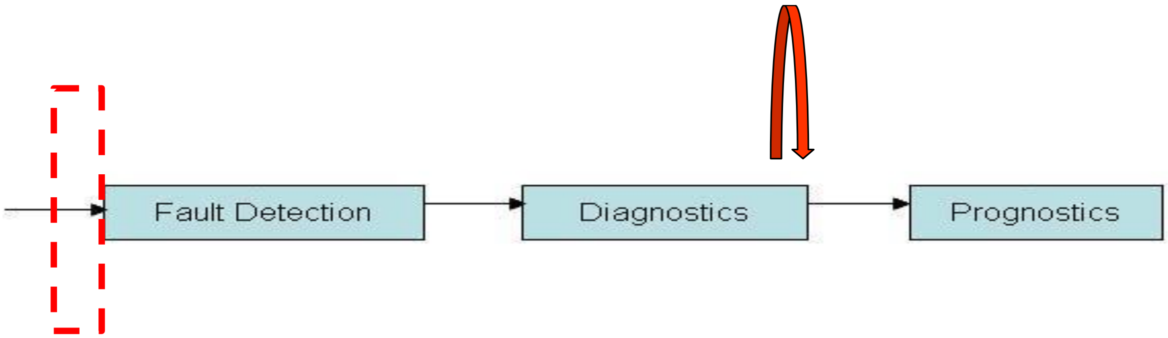

Figure 25- Extending the use of SOM to prognostics

4.5.2 Testing of Methods on KN-4073

One of the major focuses of Task-4 during the 7th quarter of this effort was to verify the overall prognostic approach on the target system. The processes for ingesting data, analysis of the data using the embedded designs and obtaining subsystem/component-level inference were tested (see Figure 26). The analyses were performed in a windows-based environment. Due to delay in generation of the data, the accuracies of the diagnostic and prognostic algorithms were not verified. However, manual adjustments to the signal processing and regression (parameter forecasting) were performed as proposed in the original V&V plans.

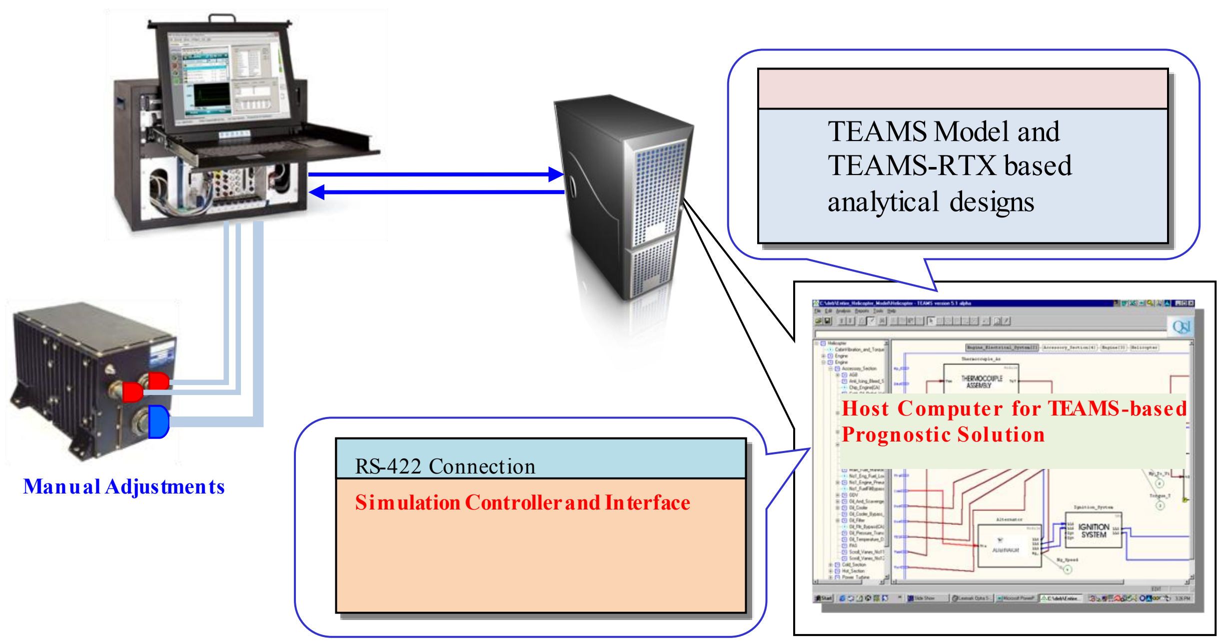

Figure 26: Verification of the Embeddable Prognostic Scheme on Target System

4.5.3 Forecasting Results on Data from KN-4073

Several types of degradation experiments were performed on the KN-4073 INS-GPS. A range of these experiments, such as accelerometer and gyro degradations resulted delayed settling time (i.e., time to reach the correct value) in heading. Short interval error/deviation forecasting and forecast for time to settle was performed during the last two months of this project. Some of the results are presented in this subsection.

Experiment on KN-4073 INS-GPS involved degradation of x, y and z axes accelerometers. This degradation caused heading errors, and it increased the time for the readings to settle to the acceptable deviation range. Deviations for 200%, 400% and 600% degradations in accelerometer are shown in Figure 27.

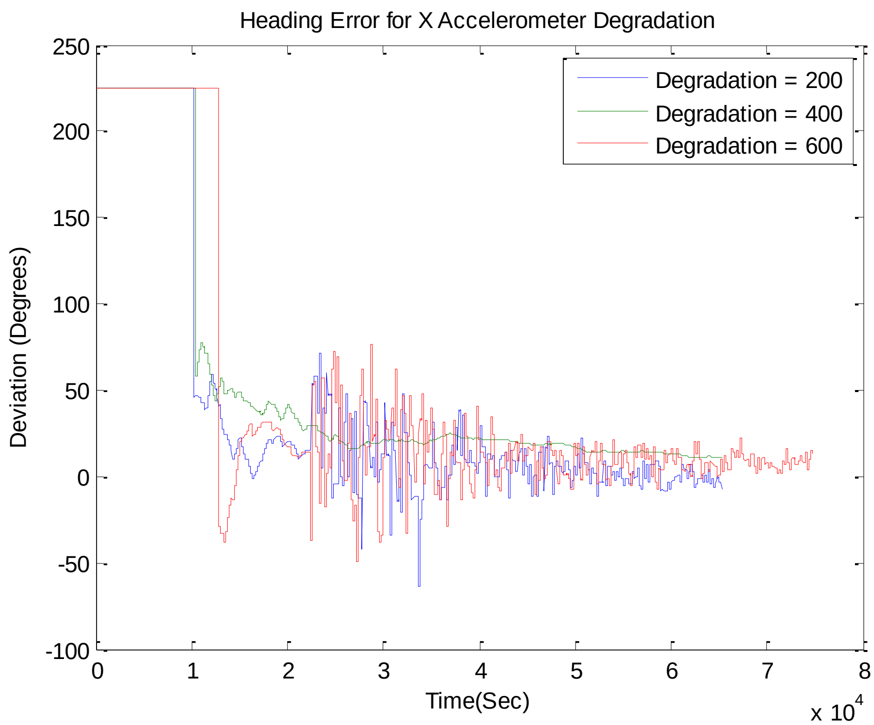

Figure 27: Heading Deviation due to Accelerometer Degradation

Two types of useful forecasting analyses can be performed using such data. The first type focuses on the short-term prediction of deviation (in True Heading estimation); while, the second type focuses on estimating the time to settling. Time-series based forecasting algorithms were used for these analyses. The results are presented in Figure 28 and Figure 29. Both Figure 28 and Figure 29 show the results for degradation in X-axis accelerometer, with degradations 200% and 400%, respectively. For the short-term degradation predictions, 30 sec prediction window was used.

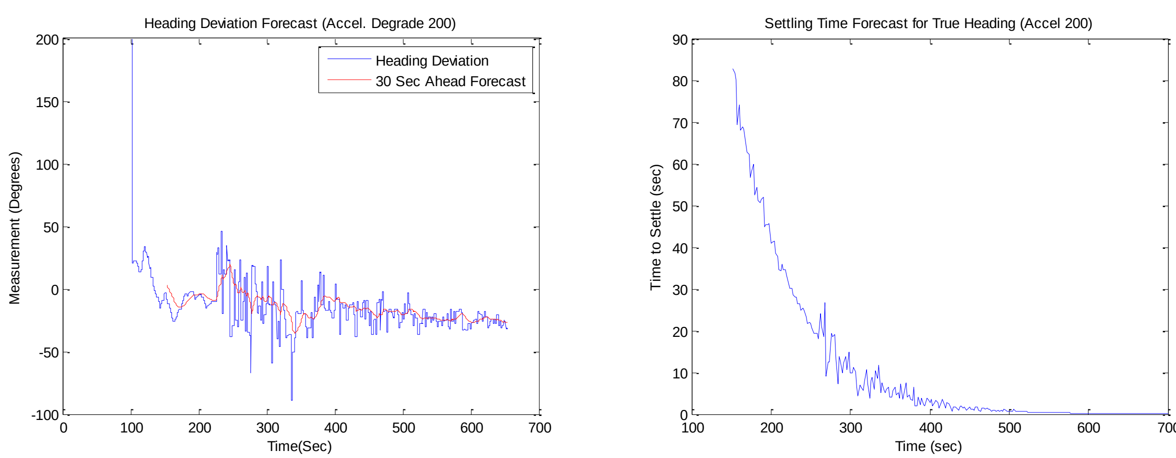

Figure 28: Deviation and Settling Time Prediction (X Accelerometer; 200% Degradation)

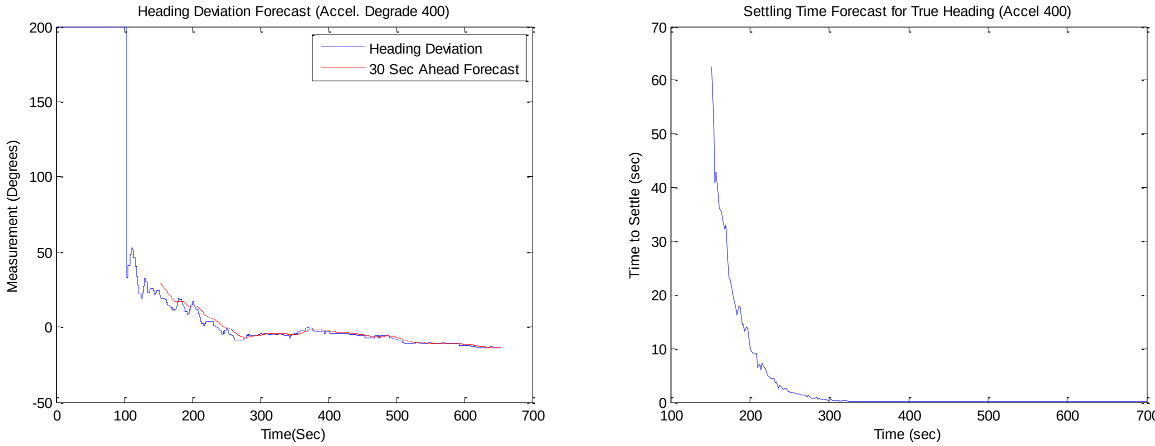

Figure 29: Deviation and Settling Time Prediction (X Accelerometer; 400% Degradation)

The usefulness of the settling time forecasting is - when the settling time exceeds a permissible time window, the INS-GPS could be considered "unfit" for an operation. Confidences are also calculated with the forecasted settling time; those values can be used for making the 'fit/unfit' decisions. A more useful forecasting could have been for the estimation of settling time of the KN-4073 that have been in use in real world and showing some degradation in performance. In essence, the settling time profiles from the degradation experiment could be used in this case.

4.6. Task-5: Automated Data Collection, Integration, Transformation, Prediction and Display Architecture Development

A major application of QSI's TEAMS Diagnostic Design and Analysis platform is as a COTS solution for health maintenance and guided troubleshooting. The TEAMS-based solution is a set of unique COTS software products that take advantage of the significant benefits and capabilities of MBR (model based reasoning) while allowing the solution to mature through feedback mechanisms, parameter update, and expert input. The modeling approach is simple, intuitive and practical, differing dramatically from other modeling approaches that require complex/expensive simulations, network constructs, mathematical equations, statistical, or state based models. The solution therefore, combines the rigor and utility of MBR with concepts of CBR (case based reasoning) and RBR (rule based reasoning). This breadth of capability allows us to provide the best COTS diagnostic solution for specific customer needs. The significance of these differences is not just technical; different approaches vary widely in their performance, in how well they address different situations, how easy they are to validate, in how easy they are to maintain, how scalable the solution is, and in what they are capable of doing within the support system. These differences translate into such important factors as deployment cost, number and duration of service calls, reduction in support costs, and above all, increased systems readiness. A short overview of the TEAMS components is given next.

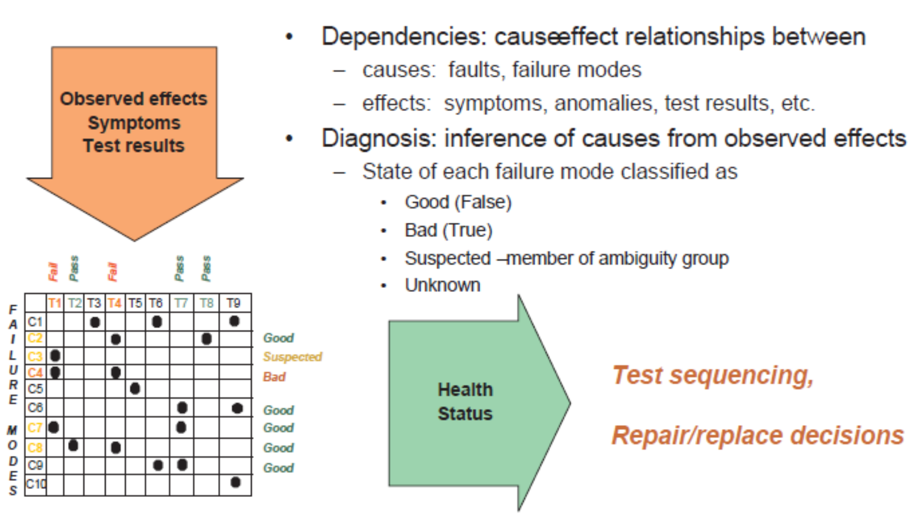

Figure 30: Use of Dependency Models for Health Inference and Diagnostics

4.6.1 QSI's TEAMS Software Suite

TEAMS Tool Set consists of four software applications: TEAMS Designer, TEAMS-RT, TEAMATE and TEAMS-RDS [40], [41], [42], [43]. Underlying these tools is the model and diagnostic data knowledge base called TEAMS-KB. The TEAMS model of a system is a dependency model that captures relationships between failure modes of the system and their observable effects. The model is created in TEAMS Designer, or imported into TEAMS Designer from other data capture environments, and then analyzed and converted into run-time versions for export to the run-time reasoners TEAMATE and TEAMS-RT. Figure 30, Figure 31 and Figure 32 illustrate the concept of dependency modeling and its use for diagnosis, the creation of dependency models in TEAMS Designer, and their use by the run-time reasoners in TEAMS-RDS.

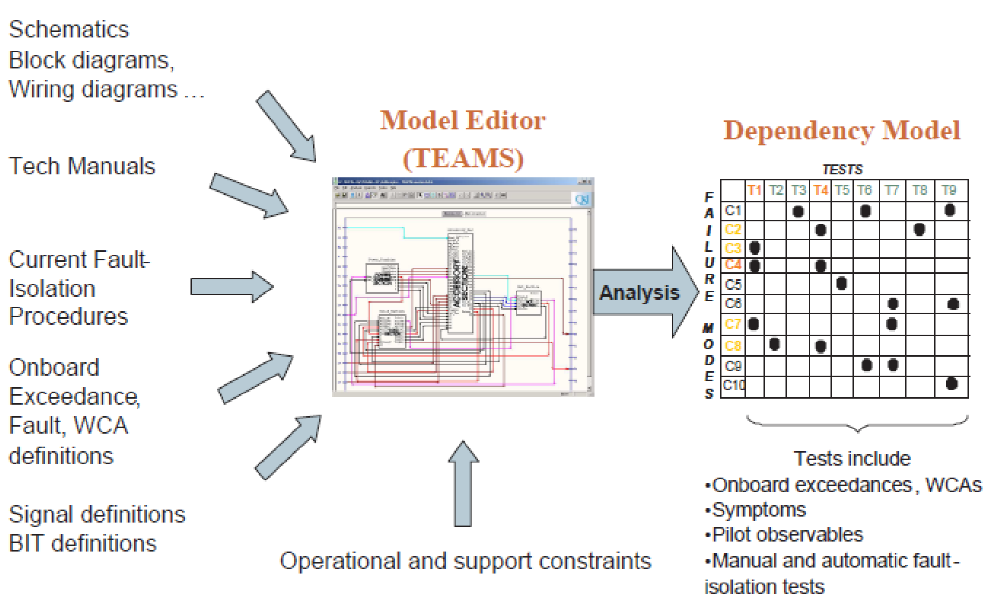

Figure 31: Creating a Dependency Model in TEAMS

The TEAMS Designer application provides a user-friendly graphical environment for developing dependency models of systems while allowing the specification of several additional practical aspects about the system that are required by the run-time inference engines to provide efficient diagnosis. It does so by allowing the modeler to specify cause-effect dependencies using a hierarchical, multi-layered (multisignal), directed graph representation of the system. In this graphical representation, the system's physical elements (subsystems, components, etc.) are represented as module nodes; the physical locations, where the measurements of the system's performance or other attributes are made for the determination of a failure or anomalous condition, are represented as test-point nodes; and the dependency relationships are represented as directed links (edges). After the model specification is complete, a reachability analysis can be performed in TEAMS to internally generate the dependency matrix model of the system subject to analysis constraints specified by the user.

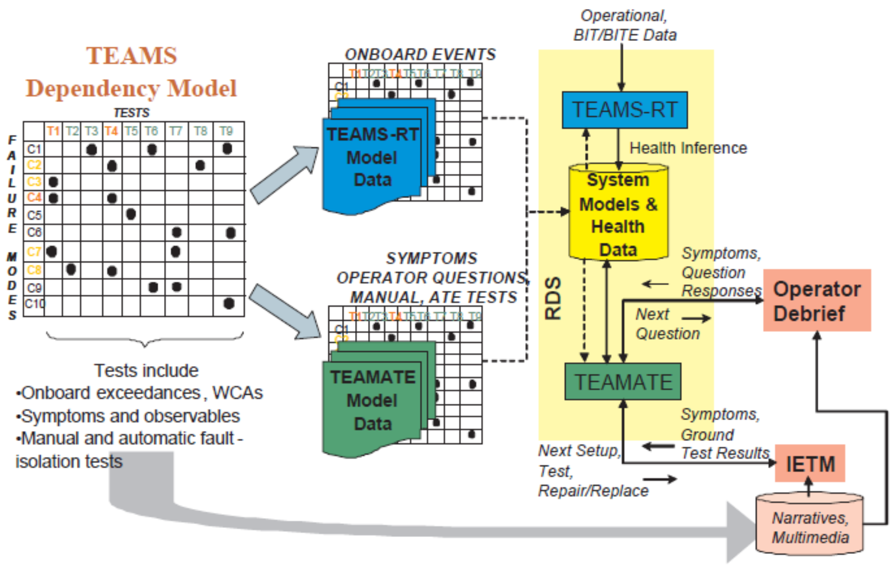

Figure 32: TEAMS Model Export to Reasoners and Presentation Interfaces

Once the dependency-matrix model is available, diagnosis becomes the process of using the dependency relationships and the observed failures or anomalies to infer their possible causes. The functional requirements of the reasoning engine that performs the diagnostic inference depend on the manner in which the observations about the system's state become available.

The TEAMS-RT inference engine processes failure events (exceedances, built-in test failures, performance anomalies, etc.), as they become available. It uses the data to infer the status of the root causes (the identification of one or more component faults). Thus, TEAMS-RT is appropriate for processing onboard data that is either received in real time or downloaded post mission/operation. The TEAMATE diagnostic reasoner not only performs inference of component health status, but also computes an optimal sequence of (active) tests that needs to be performed for fault isolation, given the current inferred health status, the allowable set of tests, and any precedence constraints on the tests. Thus, TEAMATE is appropriate for ground-based deployment where troubleshooting is performed interactively.

4.6.2 Extensions to the Information Interchange Capacity of TEAMS

The real-time component of TEAMS, the TEAMS-RT is equipped with the capability to ingest binary data (test outcomes) and disseminate health status to the external world through a network connection. An utility program "Sensor Agent", provides this functionality. This program uses the run-time model files exported from QSI's testability analysis tool – TEAMS-Designer, to seed the faults in a simulator, and send the corresponding Pass/Fail test outcomes to the TEAMS-RDS server running the TEAMS-RT diagnostic engine. The TEAMS-RT engine in the server, interprets the test outcomes sent by various Sensor Agents, and diagnoses the system(s) health. The TEAMS-RT, is one of the components of QSI's Remote Diagnostic Server (RDS) product suite. Sensor agent can obtain real-time data from a network connection using the Java MessageUtil class [44]. A simplified representation of the data ingestion and health status dissemination process is shown in **Figure 33**. Users can develop and embed their own custom TEAMS-RT clients for real-time diagnosis that will send the pass/fail test results to TEAMS-RT server.

To learn about how to develop custom sensor clients, refer to RDS Software Development Kit (SDK) from QSI.

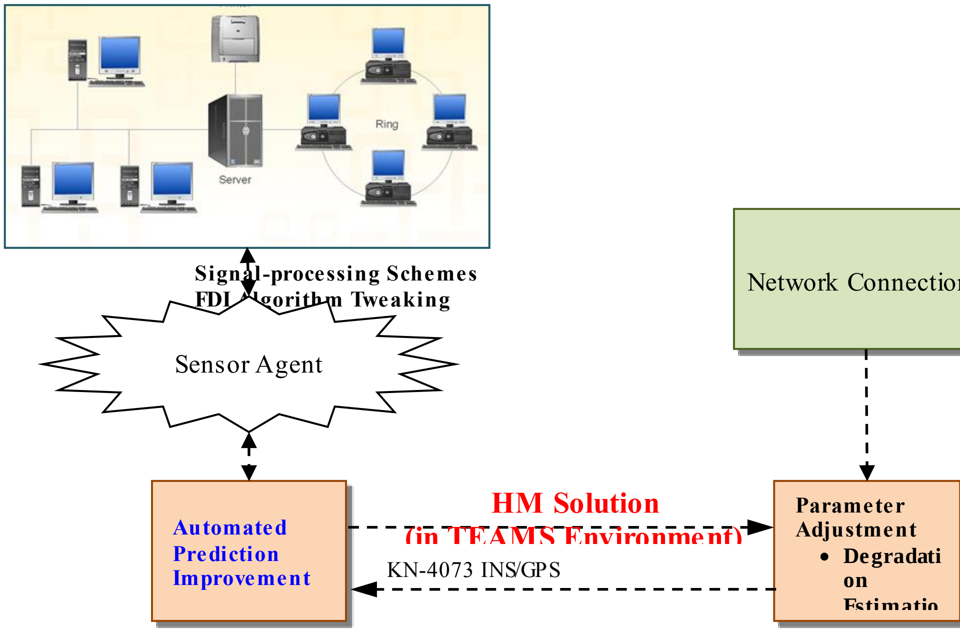

Figure 33: Simplfied Information Interchange Process

The Sensor Agent client performs the following functions.

1. Initiates a connection with the server,
2. Starts a new on-line diagnostic session for the intended system by loading the TEAMS-Designer exported run-time model for the system
3. Computes the likely pass/fail results for the user seeded faults based on the model d-matrix and the current switch modes selected,
4. Fills the RDS message structure with pass/fail test results that correspond to the inserted faults,
5. Packs the structure into a message string that conforms to RDS messaging protocol, and sends the packed string to the TEAMS-RT server,
6. Decodes the diagnosis or system health report returned by the TEAMS-RT server (which is based on the sensor results sent in Step 3) or other commands such as save, exit etc.,
7. Displays the diagnosis received in Step 6.

The existing information interchange facility in TEAMS is adequate to work with the cases where data is binary; however, most monitored data in real-world are “real valued”. To address this issue during the Nov 09 – Feb 10 period of performance, QSI worked on developing a utility that may read data from either a file or a database[‡‡]. As a result, a tool capable of reading data from CSV file (formatted as per QSI’s instruction) was developed. This tool provides point-to-point and point-to-group data ingestion capability.

[‡‡] The extended information interchange capability is originally an internal R&D effort of QSI. This SBIR contributed towards adoption of some project-goal specific options in that effort.

The utility has a wide range of compatibility and can work on Windows and number of UNIX platforms. At present, this data ingestion utility is fully functional and is undergoing testing.

4.6.3 Developing Data Ingestion Capability for KN-4073 HIL Simulation

QSI developed the RTX-client for data ingestion for this experiment. The client was built to verify that data from a real-time source (sensor scheme, this case) can be brought into the RTX environment. Transformation of data (described in Section 4.4.3.2) is automatically done as a built-in intermittent step in the data ingestion process. The KN-4073 has above 100 parameters that are monitored during simulation runs. Most of the parameters are sampled at 1Hz; however there are some parameters that are sampled at 50Hz and 100Hz as well. Therefore the data ingestion client verified the capability for ingesting large dataset with diverse sampling times. The RTX test designs that would eventually use these data were not built at this time. Those will be built during the upcoming period of performance.

5. Conclusions and Future Plan

The goal of this STTR was to develop an end-to-end deployable solution for prognostics and condition monitoring in electronic systems using data driven methods. Through this effort, QSI developed algorithms for forecasting of measurable/observable performance parameters of electronic systems. Those algorithms were implemented in Software. QSI also enriched the TEAMS-RTX test designer by improving the GUI and introducing several analytic modules for enabling forecasting algorithm implementation. For proving the accuracy and efficacy of the algorithms, QSI in collaboration with NGC performed hardware-in-the-loop (HIL) simulations on KN-4073 INS/GPS a limited set of degradation (injected) conditions. This data was used to test and verify the accuracy and usability of the forecasting techniques developed through this effort. A data-ingestion front-end was introduced to the TEAMS-RTX such that it could be deployed in real-world condition monitoring and forecasting applications for this specific system. TEAMS-RTX was tested out for performing real-time performance/condition estimation and continuous forecasting using both simulated data and data from the KN-4073 INS/GPS.

The techniques and software tools resulting from this effort are in fairly matured condition, the QSI team expects to transition these tools and techniques into deployable and commercializable products through follow-up Phase-III effort, or through some other program of DoD. In near future, QSI would try to hold discussions with the TPOC and other interested personnel in the Army, and identify the suitable approach towards putting the outcomes of this effort for use in DoD agencies.

References

[1] "RTCF User Manual", Version 2.1, Northrop Grumman Corporation, Industrial System, CA, USA, Jan 2007.
[2] Miranda Keeney, "Common Logistics Operating Environment," in Information Brief DoD Maintenance Symposium, 25 Oct 2005
[3] "Future Combat Systems Overview," Web hosted white paper on FCS, available at https://www.fcs.army.mil/whitepaper/whitepaper.html
[4] QSI's Integrated Toolset for health management, integrated diagnostics and telemaintenance, http://www.teamqsi.com/
[5] Deb, S., Pattipati, K.R. and Shrestha, R., "QSI's Integrated Toolset, Proc. IEEE Autotestcon, Anaheim, CA, pp. 408-421", 1997.
[6] S. Deb, S. Ghoshal, V. N. Malepati, and K. Cavanaugh, "Remote Diagnosis Server", in Proceedings of the IEEE Digital Avionics Systems Conference, Philadelphia, PA, Oct. 7-13, 2000
[7] Dunsdon, Jon; and Harrington, Mark. The Application of Open System Architecture for Condition Based Maintenance to Complete IVHM. Aerospace Conference. IEEE, 2008.
[8] MIMOSA, "Open System Architecture for Condition-Based Maintenance", 2006, http://www.mimosa.org
[9] Bezdicek et al., "How an Open Standard Development Process Led to an MDA-like Solution." MDA Success Story. Object Management Group. Ma, USA. available at http://www.omg.org/mda/mda_files/OSACBM.htm

[10] Walter, Bob. Penn State University/Applied Research Laboratory, The Boeing Company, and Machinery Information Management Open Standards Alliance (MIMOSA). Open Systems Architecture for Condition-based Maintenance (OSA-CBM) - Primer. MIMOSA, 2006.
[11] ISO TC 108/SC 5, International Organization for Standardization. Condition monitoring and diagnostics of machines - Data processing, communication and presentation - Part 1: General guidelines. Geneva, Switzerland: ISO, 2003.
[12] ISO TC 108/SC 5, International Organization for Standardization. Condition monitoring and diagnostics of machines – Data processing, communication and presentation – Part 2: Data processing. Geneva, Switzerland: ISO, 2007.
[13] L. Ramon, "Space Efficiency of XML." Science Direct 46 (2004): 753-59, 12 Apr. 2004, http://www.sciencedirect.com, (Accessed 22 July 2009).
[14] ECMA International, "Standard ECMA-334, C# Language Specification", 2006, http://www.ecmainternational.org/publications/standards/ecma-334.htm
[15] ECMA International, "Standard ECMA-335, Common Language Infrastructure (CLI)", 2006. http://www.ecma-international.org/publications/standards/Ecma-335.htm
[16] ISO, "ISO/ IEC 23270:2003 - Information technology -- C# Language Specification", 2003, http://www.iso.org
[17] ISO, "ISO/IEC 23271, Common Language Infrastructure", 2006, http://www.iso.org
[18] L. H. Chiang, E. L. Russell, and R. D. Braatz, "Fault Detection and Diagnosis in Industrial Systems," Springer, London, 2001, ISBN: 1 852 33327 8.
[19] H. Wang, Z. Song, and P. Li, "Fault Detection Behavior and Performance Analysis of Principal Component Analysis Based Process Monitoring Methods", American Chemical Society, Vol. 41, pp. 2455–2464, 2002.
[20] R. Duda, P. Hart, and D. Stork, "Pattern Classification," Wiley Press, New York, 2001.
[21] G. Taguchi, S. Chowdhury, and Y. Wu., "The Mahalanobis–Taguchi System," New York: McGraw-Hill. 2001.
[22] Bezdicek et al., "How an Open Standard Development Process Led to an MDA-like Solution." MDA Success Story. Object Management Group. Ma, USA. available at http://www.omg.org/mda/mda_files/OSACBM.htm
[23] Vachtsevanos, G., "Performance metrics for fault prognosis of complex system", In Proceedings of IEEE AUTOTESTCON on Systems Readiness, pp. 341- 345, Sept. 2003
[24] Khiripet, N., Vachtsevanos, G., Thakker, A., and Gailie, T., "A New Confidence Prediction Neural Network for Machine Failure Prognosis", In Proceedings of Intelligent Ships Synposium IV, Philadelphia, PA, April 2-3, 2001
[25] C-MAPSS Engine Simulator Data, available at https://dashlink.arc.nasa.gov/data/c-mapss-aircraft-engine-simulator-data/
[26] L. Ljung, System Identification: Theory for the User (New Jersey: Prentice-Hall, 1987)
[27] Hayes, Monson H. (1996). "9.4: Recursive Least Squares". Statistical Digital Signal Processing and Modeling. Wiley. p. 541. ISBN 0-471-59431-8.
[28] Sui Ruan, Yunkai Zhou, Feili Yu, Krishna Pattipati, Peter Willett and, Ann Patterson-Hine, " Dynamic multiple-fault diagnosis with imperfect tests," IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, Vol. 39 , Issue 6, pp: 1224-1236, Nov 2009
[29] T. Kohonen, E. Oja, O. Simula, A. Visa, J. Kangas, "Engineering Applications of the Self-Organizing Masp", Poceedings of the IEEE, Volume 84-10, October 1996.
[30] T. Kohonen. Self-Organizing Maps. Ed. T. S. Huang and M. R. Schroeder. New York: Springer, 2000.
[31] N. Gebraeel, M. Lawley, R. Liu, V. Parmeshwaran, Residual life predictions from vibration-based degradation signals: A neuralnetwork approach, IEEE Transactions on Industrial Electronics 51 (2004) 694–700.

[32] R. Huang, L. Xi, X. Li, C. Richard Liu, H. Qiu, and J. Lee, "Residual life oredictions for ball bearings based on self-organizing map and back propagation neural network methods", Mechnical Systems and Signal Processing, v. 21, pp. 193-207, 2007

[33] Y. Chai, W. Dai, S. Li, and Z Zhang, "A Self-Organizing Map Method for Optical Fiber Fiber Fault Detection and Location", Lecture Notes in Computer Science, v. 3498, pp 470-475, 2005

[34] M. Cottrell, P. Gaubert, C. Eloy, D. Francois, G. Hallaux, J. Lacaillle, and M. Verleysen, "Fault Prediction in Aircraft Enginers Using Self-Organzing Maps", Lecture Notes in Coputer Science, V. 5629, pp 37-44, 2009

[35] R. Tinos, and M. Terra, "Fault Detection and Isolation in Tobotic Manipulators and the Radial Basis Function Network Tained by the Kohonen's Self-Organizing Map", IEEE Proceedings of the 5th Brazilian Symposium on Neural Networks, pp. 85-90, 1998

[36] M. Wong, L. Jack, and A. Nandi, "Modified self-organizing map for automated noverly detection applied to vibration signal monitoring", Mechanical systems and Signal Processong, v. 20, pp 593-610, 2006

[37] B. Yang, T. Han, and L. An, "ART-KOHONEN neural network for fault diagnosis of rotating machinery", Mechanical Ssytems and Signal Processing, v. 18, pp. 645-657, 2004

[38] A. Ypma, and R. Duin, "Novelty Detection Using Self-Organizing Maps", Progress in Connectionist-Based Information Systems, vol. 2, pp 1322-1325.

[39] H. Qiu, J. Lee, J. Lin, G. Yu, "Robust performance degradation assessment methods for enhanced rolling element bearing prognostics", Advanced Engineering Informatics, v. 17, pp.127–140, 2003.

[40] S. Deb, S. Ghoshal, V. N. Malepati, and D. L. Kleinman, "Tele-diagnosis: Remote monitoring of large-scale systems," In Proceedings of IEEE Aerospace Conference, 2000.

[41] S. Deb, S. Ghoshal, and K. Pattipati, "Multisignal modeling for diagnosis, FMECA, and reliability," In Proceedings of IEEE Systems, Man and Cybernetics Conference, 1998.

[42] S. Ghoshal and A. Ghoshal, "Reusable test library - design and implementation," In Proceedings of UTECA conference, 1998.

[43] S. Ghoshal, R. Shrestha, A. Ghoshal, V. Malepati, S. Deb, and K. Pattipati, "An integrated process for system maintenance, fault diagnosis and support" In Proceedings of IEEE Aerospace Conference, 1999.

[44] Java MessageUtil class, online documentation: http://www-mipl.jpl.nasa.gov/vicar/core380/html/javadoc/jpl/mipl/pgs/ms/package-summary.html

[45] 754-2008: IEEE Standard for Floating-Point Arithmetic, Issue Date: Aug. 29 2008, pp.1 – 58,