Application of the Automated Condition Based Maintenance Checking System for Aircrafts

Mahindra IMADABATHUNI\textsuperscript{a}, Pradnya JOSHI\textsuperscript{a}, David HE\textsuperscript{a}, Mohammed AL-KATEB\textsuperscript{b}, Eric BECHHOEFER\textsuperscript{c}

\textsuperscript{a}Department of Mechanical & Industrial Engineering, The University of Illinois at Chicago, Chicago, IL 60607
\textsuperscript{b}Department of Computer Science, The University of Vermont, Burlington, VT 05405
\textsuperscript{c}Goodrich Sensors and Integrated Systems, Vergennes, VT 0549

Abstract

Condition Based Maintenance (CBM) systems have evolved as an effective fault detection mechanism in aircrafts by reducing the costs associated with unscheduled maintenance. CBM systems help maintainers to detect and manage the condition of aviation system components and take maintenance actions when there is evidence of need. In this paper we describe the application of a software prototype, which is an automation of the CBM practices, in generating the maintenance reports that are deemed essential for continuous improvement under CBM. We explain the procedure in generating the reports using the software prototype, named the Automated Condition Based Maintenance Checking System (ACBMCS), which is configurable across different platforms and can perform new operations without having to modify the existing source codes. We illustrate the building of the configuration information, required to generate these reports, using XML. We demonstrate that the developed system has the functional capabilities essential to implement CBM on any aircraft. We generate WOW-Status Check reports, Fault BIT reports, Exceedances reports, Vibration Diagnostics reports, Rotor Track and Balance (RTB) reports, Engine Performance reports and Missing Data reports.

Key words: Condition-based Maintenance, XML Schema, Software Configuration Design

1 Introduction

Condition Based Maintenance (CBM) is an important step towards providing Reliability Centered Maintenance (RCM) to avoid failures in aviation equipment. CBM is defined as a set of maintenance processes and capabilities derived primarily from realtime assessment of weapon system condition obtained from embedded sensors and/or external test and measurements using portable equipment [2]. The specific technical and functional requirements of CBM are detailed in [2]. CBM can be highly reliable in allowing maintainers to identify the condition of different components and take corresponding maintenance action only when a need for maintenance is established. The advantages of CBM implementation are evident and are listed as its goals [2]:

Email addresses: mimada2@uic.edu (Mahindra IMADABATHUNI), pjoshi5@uic.edu (Pradnya JOSHI), davidhe@uic.edu (David HE), malkateb@cs.uvm.edu (Mohammed AL-KATEB), Eric.Bechhoefer@goodrich.com (Eric BECHHOEFER).
(1) reduction of burdensome maintenance tasks required to assure continued airworthiness
(2) increase of aircraft availability
(3) improvement of flight safety
(4) reduction of sustainment costs

Health and Usage Monitoring Systems (HUMS) have evolved in parallel with the concepts of CBM. They are the elementary forms of fault detection, through processing of signals from sensors, in aircraft troubleshooting and inspection. The sensors and signal processing provide information regarding a parts condition and the need for maintenance action. This is the essence of CBM and the HUMS.

There has been steady evolution of capability to monitor the performance and conditions of major systems and subsystems of rotary wing aircraft since the 1970s, with the first installation of the HUMS onboard helicopters in the 1990s [2]. As systems became more complicated, numerous parameters were included to the list of things to be monitored. Thus development of HUMS gained much importance to enable maintainers to use the systems effectively, efficiently and with ease. CBM involves collection of data from sensors, and the processing, analysis and correlation of that data to standards to determine maintenance action. Data is the essential core of the CBM and massive amounts of data is continuously collected and processed by the HUMS. Thus the need for an efficient automation of the system is established.

The objective is the development of a tool that automates the process of identifying the faulty components on which maintenance is necessary and which allows maintainers to perform different and new analysis without having to modify the application source code. Thus the software, which is driven by configuration, eliminates the need for new software releases, reducing cost. The fact that new software releases are typically expensive to test, is reason enough to build a generic configuration-driven data mining tool ACBMCS. Also, the framework presented in this thesis is a novel concept and will find huge application in the field of CBM.

The functional requirements for any CBM system, in the aviation industry, essentially include engine monitoring, drive train monitoring, structural monitoring and exceedance monitoring [2]. Our goal is to demonstrate that ACBMCS is capable of performing all these functional requirements. We seek to verify ACBMCS functionalities and apply it successfully to generate WOW-Status Check reports, Fault BIT reports, Exceedances reports, Vibration diagnostics reports, Rotor track and balance (RTB) reports, Engine performance reports and Missing data reports for different aircrafts.

In Section 2, we introduce the software prototype and briefly describe the novel framework on which it is built. In Section 3, we describe the different maintenance reports generated and explain the procedure of building the configuration files required to generate these reports.

2 The System

2.1 XML-Based Configuration Design

The primary requirement of ACBMCS is generality. The identification of the different operations to be performed by the system drives the design of a generic system structure. See Figure 1. The software requirements specifications ensure the proper functioning and maintenance of the software. Once the system structure and the requirements are established, the system is modeled using XML. An XML-based design addresses all the design challenges. The XML-Based Design involves design of the XML Schema and the XML files. The XML Schema includes information regarding the organization of the elements, sub-elements and attributes which is in accordance with the structure of the system. The different restrictions are assigned on each of the elements, sub-elements and attributes to ensure data integrity. The XML files hold the information and are validated against the Schema to check for discrepancies.

The XML schema models the following key facts:

(1) The same aircraft may be associated with more than one analysis
(2) The same analysis may be applied to different aircrafts
(3) For each aircraft, and for each analysis associated with this specific aircraft, there are parameters that should be considered for such aircraft-analysis combination.
Fig. 1. Structure of the System

(4) Every parameter is associated with an algorithm that has to be executed for obtaining maintenance decisions
(5) A report is displayed describing the maintenance decision from every algorithm

Thus system design essentially involves extensive modeling of the above five elements:

(1) Aircraft
(2) Analysis
(3) Parameter
(4) Algorithm
(5) Report

The design begins with assigning constraints on every element of the configuration, which is aircraft, analysis, parameter, algorithm or report, by means of the XML schema. The configuration is defined by these five elements and the configuration information is stored in XML files in accordance with the structure specified by the Schema. The details of the system design can be found in [1].

2.2 Software Architecture

The ACBMCS software architecture is composed of three connected layers

(1) Storage Layer: The storage layer is the lower lever layer which stores configuration metadata that drives the processing of the systems, as well as source data on which analysis is being done. The configuration metadata (i.e., information representing aircrafts, analysis, parameters, algorithms, and actions report) is stored as a collection of XML documents. The source data, on the other hand, comes in the form of Raw Data Files (RDF) and Activity Data Files (ADF). An RDF is a collection of aircraft health data for a single aircraft operation, whereas an ADF is a set of indexes, to an RDF file, that provides a performance optimized approach of retrieving source data of a single aircraft operation.
(2) Extraction Layer: This layer carries out the extraction of data from the source files. This data extraction is derived by the content of configuration metadata.
(3) Processing Layer: The processing layer is responsible to execute the mining algorithms, based on the configuration information, against the corresponding data retrieved from source data files. The details can be found in [1].

2.3 Software Features and Usage

The user interface is designed such that the installation and usage complexity of the system is reduced to minimum. When the ACBMCS application is launched, a main form giving the user an option to run the software in AUTO mode or MANUAL mode is displayed (Figure 2).

In the AUTO mode, the software generates the reports automatically on all the adfs in the default directory. In the MANUAL mode, the system engineer is required to perform these basic steps to generate maintenance reports:
(1) Load the XML Schema
(2) Load XML files which are the storage layers of meta-data
(3) Validate XML files with XML-Schema
(4) Load the configuration
(5) Select the aircraft
(6) Select the analysis to be done
(7) Select the parameters for the analysis
(8) Load the ADF files and select the files for report generation. This can be done in three ways. Either by selecting all files from a directory or by selecting arbitrary set of files or by selecting files within a date range.
(9) Generate Report. Select to display only faulty items or non-faulty or both.

The configuration data can be changed (generic application) depending on the requirement. The application automatically loads any previously loaded configuration files upon selection of the manual mode or gives the user an option to reload configuration files. Once the configuration is loaded, the analysis to be performed on the required aircraft is selected. Then the parameters associated with the analysis are displayed. Once the parameters are selected, the algorithm associated with the analysis and the report of the corresponding algorithm are displayed for the user information. See Figure 3.

3 Reports Generation

3.1 Maintenance Reports

ACBMCS is capable of generating the following reports:

(1) Fault BIT Report: Built-in-Test (BIT) is an invaluable component of modular, embedded systems that are used for critical applications such as avionics mission systems, sensors, and weapons. BIT provides a level of confidence in the correct operation of each module at both power-up and during normal operation. It is increasingly important to evaluate BIT’s performance as it provides fault finding as a means to aid in system assembly, test and maintenance.

(2) Exceedances Report: The HUMS Exceedance Processing consists of monitoring events to display information to the crew. A monitored event is raised when a particular condition or set of conditions becomes valid or true. The monitored event is designated as an exceedance event and the crew notification is given [3]. The exceedances report is generated based on the threshold algorithms.
detailed in [3]. The HUMS Exceedance Processing and Monitoring requirements are defined in [4]. The requirements of the system define the exceedance conditions. The exceedance definitions can be found on Appendix IV of [5]. These exceedance definitions form the set of conditions for monitoring an event.

(3) Vibration Diagnostics Report: Vibration Diagnostics is a measure of either noise or mechanical vibration. Vibration diagnostics may be used to detect mechanical failure in machinery having components such as gears, bearings, or rotors.

(4) Rotor Track and Balance Report: Rotor track and balance is the process of smoothing vibrations in the airframe, which are caused by the main rotor. The main rotor is not the only rotating assembly of concern in a helicopter; there are others such as the tail rotor assembly, drive shaft assemblies, and oil cooler fans.

(5) Engine Performance Report: This report concerns all HIT checks, Power checks, and the ability to do real time engine performance checks.

(6) WOW Switch Check: Weight-on-Wheels (WOW) switch, activates a number of relays that control several flight control systems like landing gear wheels and brakes, electrical power system, AC power generation, flight environment systems, engine power control, nose wheel steering system etc. This WOW switch is subjected to a lot of vibrations in the aircraft’s operating environment; hence it is important to check the health condition of WOW-Switch.

3.2 Development

Development of these reports using the ACBMCS involves identification of the configuration specifications required to generate these reports. These details are then fed into the XML configuration files. This information is later retrieved and processed by the generated dynamic code to display the required report. The development of the dynamic code in the .NET framework to generate the above mentioned reports leads to the categorization of the reports. Identification of the different types of reports is the most crucial step in the development of a software which is configuration-driven that is generic.

The reports list the type of analysis being performed, the aircraft type, the tail number and unit number of the aircraft and the report action. The need for a classification arises because different analyses have different number and type of algorithms to be executed before the generation of a final report. The different types of reports generated by the ACBMCS are listed below:

(1) Type-1: When there is a single algorithm to be executed for one type of analysis, the report will be same as that described above.

(2) Type-2: When there are different algorithms, involving different parameters, to be executed for a single type of analysis, the report shall list the name of the parameter along with the report action.

(3) Type-3: For certain analysis, the count of failures in all the parameters shall be listed.

(4) Type-4: When there are more than one source files available for executing the required algorithm, the ACBMCS shall allow selection of multiple source files and generate all the report actions in one single file.

(5) Type-5: Finally, whenever data for a parameter of interest is missing the report action shall text out: DATA MISSING against the parameter name in any of the above types of reports. This covers the requirement of the generation of the Missing Data Report.

To distinguish between the requirements for every analysis, a “categoryID” is specified for every analysis type in the xml file. The “categoryID” is a code formed from binary values. By changing the “categoryID” the user may change the desired type of output.

3.3 Building the Configuration

(1) Building the "analysis" Configuration File:

```
  <Analysis anaID="Axxxxx" categoryID="xxxx"/>
</AnalysisSets>
```

(AnalysisSets: Define all possible "Analysis" types under this main element)

(Analysis: Every analysis is given a specific ID called the "anaID" obeying the restriction pattern
defined in the schema. The "categoryID" is a binary code which distinguishes the requirements of
different analysis as follows:
First binary digit: Multiple parameters for same analysis
Second Binary digit: Multiple values for same parameter
Third Binary Digit: 1-Check latest value 0- Check all values
Fourth Binary Digit: Display count of failures1-True 0-False)

<name>xxxxxxx</name>
(name: Input the name of the Analysis)
</Analysis>
</AnalysisSets>

(2) Building the "aircraft" Configuration File:
(Aircrafts: Define all the possible "Aircraft" types under this main element)
<AirCraft airID="AIRxxx">
(Aircraft: Assign an "air ID" to every aircraft following the pattern defined in the schema)
<airname>xxxxx</airname>
(airname: input the aircraft name)
<analysistypes>
(analysis:types: define all possible analyses that may be performed to determine the health condition
of the aircraft)
<assignment3 analysisID="Axxxxx"/>
(assignment3: analysis are assigned to the aircraft by referencing using their "analysisID")
<assignment3 analysisID="Axxxxx"/>
</analysistypes>
</AirCraft>
</Aircrafts>

(3) Building the "parameter" Configuration File:
(Parameters: Define all possible parameters required to perform an analysis under this element)
<Parameter paramID="Pxxx">
(Parameter: All possible parameters are defined by this sub-element and every parameter is given a
"paramID")
<paramname>xxxxxxx</paramname>
(paramname: input the name of the parameter)
<datatype>xxxxxx</datatype>
(datatype: input the location of this parameter)
<usedAt>
(usedAt: assign which aircraft-analysis combination will use the parameter)
<assignment analysisID="Axxxxx" aircraftID="AIRxxx"/>
</usedAt>
</Parameter>
</Parameters>

(4) Building the "algorithm" Configuration File:
(algorithms: define all possible algorithms to be executed to perform the different analyses)
<algorithm algoID="ALxxx">
(algorithm: assign an "algoID" to every algorithm type defined)
</algorithm>
<assignment2 paramID="Pxxx"/>
(first input the parameters used in the algorithm by using their "paramID")
</algorithm>
</algorithms>
(algoinfo: gives the details of the algorithm)
<condition>
<define the condition>
<SimpleCondition>
<define if the algorithm has a simpleCondition type (IF "condition") or if its complex (IF "condition1" "logical operator" "condition2"))
<operand>
<operate>Pxxx</operate>
(the operand may be a string, decimal or integer)
</operand>
<operator>==</operator>
(input one of the operators =,<=,>=,!=,>,<)
</Operand>
<operate2>0</operate2>
(the operand may be a string, decimal or integer)
</Operand>
</SimpleCondition>
</name>
</condition>
<action>
<define the action or result>
<name>
<actions>OBS=RED</actions>
</name>
</action>
<algoinfo>
<algoinfo>
<SimpleCondition>
<operand>
<operate>Pxxx</operate>
</operand>
<operator>==</operator>
</Operand>
<operate2>1</operate2>
</Operand>
</SimpleCondition>
</name>
</action>
<name>
<actions>OBS=GREEN</actions>
</name>
</algoinfo>
</algorithm>
</algorithms>

(5) Building the "report" Configuration File:
<report repID="RIDxxx"> 
(report: define all the different types of reports to be generated under this element) 
<for> 
(assign algorithm types for which the report type is applicable) 

(gives the details of the report)

(name)

(SimpleCondition)

(operand)

(the operand may be a string, decimal or integer)

(Operator)

(input one of the operators =,<,>,>=,<=,<)

(Operand)

(the operand may be a string, decimal or integer)

(BIT : PASS!)

(define the action or result)

(BIT: FAIL!)
4 Results

We input the configuration files and run ACBMCS to generate the reports. See Figures 4, 5, 6, 7, 8, 9.

5 Conclusions and Future Work

The software has been successfully used to generate reports for WOW-Switch check, Fault BIT analysis, Exceedance Monitoring, Vibration Diagnostics, Rotor Track and Balance diagnostics, Engine Performance and Missing Data. All reports are organized by Unit, Aircraft Type and Model, with a single report being
generated for multiple ADF/RDF files. Display of the count of failures, when required, is also given in the report. Three different modes for loading the adf/rdf files are presented in the software application, which adds flexibility to the user. Reports can be generated for only faulty items, only non-faulty items, or both.

This software opens an avenue for several future works including, for instance:

1. Mechanical Diagnostic Report that involve analysis of HI data.
2. Reports with more complex data mining algorithms, visual configuration tool to provide an easy-to-use graphical user interface for the user to configure the XML files.
3. Optimization Techniques like Stream processing, random sampling; reservoir sampling; stratified sampling; Temporal Data Analysis

6 Acknowledgements

This paper is based upon work supported by Mike Wiley, Program Manager, UTILITY HELICOPTER, US ARMY and Goodrich Corporation.

References