A REVIEW OF PHM SYSTEM’S ARCHITECTURAL FRAMEWORKS

Surya Kunche, Chaochao Chen, Michael Pecht
Center for Advanced Life Cycle Engineering
Department of Mechanical Engineering
University of Maryland
College Park, MD 20742

Abstract – The need for a Prognostics and Health Management (PHM) system is propelled by increasing demand for condition based maintenance for systems to reduce cost of maintenance and mitigate risk. The use of multiple algorithms for PHM for a variety of systems presents challenges for PHM system developers in terms of integration and interfacing of various components, including hardware and software. A PHM system comprises of several elements including sensors, computing hardware and software algorithms for fault detection, diagnostics, prognostics and decision support. Thus there is a need for an architectural framework to help system developers and integrators for faster system development and deployment.

An architectural framework provides a blueprint to enable the constituent subsystems to serve the overall purpose of the system. This also facilitates the interoperability of PHM systems with various applications. It provides a holistic view of the system for system developers and thereby forms the basis for building a system or serves as a guide for system modeling, integration and testing. This paper reviews the architectures proposed in literature for a PHM system. The architectural frameworks proposed for various PHM applications are also discussed and reviewed. The reviewed frameworks integrate various functionalities of the PHM system including data acquisition, signal processing, feature extraction, anomaly detection, diagnostics, prognostics and decision support. The architectures are categorized as either functional or physical architectures. The advantages and shortcomings of these proposed frameworks are also discussed.

1. INTRODUCTION

Prognostics and health management (PHM) is an enabling discipline consisting of technologies and methods to assess the reliability of a product in its actual life cycle conditions to determine the advent of failure and mitigate system risk [1].

Prognostics is a process of correlating precursor parameter with the system performance parameters to estimate the Remaining Useful Life (RUL) for a product based on historical conditions [2][3]. The RUL being a measure of degradation process is estimated by extrapolation based on analysis of past observations; this estimate also has an innate uncertainty associated with its predicted value [4].
Health management involves safeguarding the system from potential failure and malfunction of component or the system a whole. This is achieved by appropriate scheduling of maintenance or by mitigation of failure [5]. Health management makes use of RUL estimates for appropriate scheduling of maintenance activities. It also prevents early replacement of components by enabling predictive maintenance rather than preventive maintenance strategies which lead to wastage of resources.

Architecture is the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design evolution [6]. Essentially architecture serves as a framework for building a system by helping the system developers to integrate various components (sub-systems) of the software and could also help project managers achieve better project management during the process of systems building by proving them with system requirements and the scope of work required for the system.

During the past decades, PHM technologies, such as data processing, feature extraction, fault diagnosis and failure prognosis, have become more mature. Numerous new algorithms and techniques for PHM have been published in various fields [7]-[10]. But very few studies on PHM system architectures have been reported. With the rapidly increasing complexity of the PHM system, modular and flexible system architectures are needed to seamlessly integrate all of the PHM components. The industry and military are eager to acquire such system architectures that enable the PHM components to be easily deployed and maintained in target systems, meanwhile, to be implemented in real time and on board.

The following section provides a discussion on the basics of architecture. Various proposed architectural frameworks for a PHM system are discussed and reviewed. The requirements of a PHM architecture as documented in the ISO standard ISO 13374-2 are also discussed in the next section.

2. DISCUSSION

A system can typically be described as a combination of two building blocks, namely functional architecture and physical architecture [11]. The functional architecture represents the functionalities of the system as blocks and the data flow between these functional blocks. When a system is decomposed into different functionalities for achieving the system’s goal, the functional architecture provides a relationship between these various blocks and shows the interactions between them. The physical architecture demonstrates the relationship between various hardware resources of the system (such as the computing resources). It shows the components in the various layers of the architecture and the physical distribution of the components in the system. The functional and physical architectures are developed in parallel with close interaction between them to form a meaningful operational system from system requirements (operational concept). The operational system is a combination of functional and physical architectures. The Figure 1 below shows the relationship described above.
The architecture provides a fundamental understanding of the system components. It enables the system developers to form a basis for system integration and qualification before deployment of the system. PHM being a multi-disciplinary domain, various functional blocks of the system are developed by engineers from different domains of research. Hence, there is an even more need for a generic architectural framework to help in integration of the system thereby their testing and qualification before final deployment.

Some of these architectural frameworks are discussed in this section. The architectures reviewed are classified as either functional or physical architectures of the PHM system.

2.1 PHM Functional Architectures

The following architectures show the functional overview of the system. These architectures describe the flow of the information in the systems.

2.1.1 Open System Architecture – Condition Based Maintenance (OSA – CBM)

The OSA-CBM provides a functional architecture for the CBM. The OSA-CBM has been proposed by the maintenance information open system alliance (MIMOSA) which is an alliance for operation and maintenance system solution providers and end user companies. The architecture provides a prototype framework for integration of various hardware and software components for Condition Based Maintenance (CBM). The architecture consists of six functional layers [12] and is as shown in Figure 2.
i. **Sensor Module:** The sensor module provides the system with sensor or transducer data. It essentially measures observable parameters of a system.

ii. **Data Acquisition:** It samples data from the sensor modules.

iii. **Data Manipulation:** This layer essentially performs the data manipulation function. It transforms the received signal to usable forms. This block is responsible for providing high quality data without losing vital features of the sensor signals to the lower layer. This process may include tasks like de-noising, signal frequency and wavelet analysis by using Fourier transforms, wavelet transform etc.

iv. **State Detection:** It receives the transformed data and compares with past expected values and operation limits to generate alerts. Fault identification is an example of condition monitoring.

v. **Health Assessment:** This layer performs the diagnostic function. It assesses the data from condition monitoring and prescribes if the health in a particular component or sub-system or the system is degraded. Some of the diagnostic methods used include signal processing, machine learning, and fusion methods.

vi. **Prognostics:** It estimates the Remaining Useful life based on past machine conditions, meanwhile, considering future usage trends.

vii. **Advisory Generation:** It generates recommended actions for system maintenance and how to run the system (system reconfiguration) during the life cycle.

The goal of OSA-CBM is to promote interoperability which implies exchange of information between various layers and ease of integration with various sub-components of the system and bridge the gap between personnel from varied research backgrounds. To achieve this, the layers communicate by a set of fourteen defined functions which serve as a communication interface. Very tight coupling (i.e. highly hierarchical
architectures) among various layers may render them un-interoperable and un-interchangeable for other applications or systems. This architecture was also adopted by the standard ISO 13374-2-2007: Condition monitoring and diagnostics of machines [13]. This standard identifies the fact that various software currently available for condition monitoring cannot exchange information and cannot operate in a plug and play manner without extensive integration efforts. Hence, ISO 13374-2-2007 collection of standards enable condition monitoring software to be platform or hardware independent by using some standard communication protocols that are platform independent.

2.1.2 Diagnostic and Prognostic Framework for CBM

Baruah et al [14] have proposed a Diagnostic and Prognostic Framework (DPF) for CBM. The architectural framework proposed by Baruah et al is as shown in Figure 3. The DPF performs diagnostics and prognostics and it provides inputs for the Decision Support System (DSS) which recommends appropriate maintenance actions. The inputs for the DPF typically consist of extracted features from the sensor signals. Based on the sensor signals it provides the DSS with prognostic and diagnostic data. Principal Component Analysis (PCA) was used as a dimensionality reduction technique. The authors used kernel density estimation based clustering techniques on the two most dominant principal components for clustering modes of operation of equipment. Classification is also performed on this two dimensional principal component data. Envelope detection involves calculating mean and standard deviation of healthy data and checking whether current measurements deviate significantly from healthy data. The diagnostics consists of three independent analyses based on classification, feature or signal enveloping and velocity threshold based diagnostics. The prognostics used a pre-determined degradation model to predict the degradation of the system.

The authors demonstrated the performance of the system by compiling the algorithm in Matlab as a standalone application. The authors demonstrated the functioning of the algorithm on rotating bearing subjected to accelerated degradation.

A similar PHM framework has been deployed at the Ford’s manufacturing plant for monitoring electromechanical devices. Very few false alarms were observed by using this system. Online monitoring helps in vehicle health monitoring in field and it can be put to best use in the fleet monitoring. The middleware architecture which enables communication among different functional blocks is not well addressed. The biggest drawback of this proposed system is that it only provides a functional overview of the system and does not provide future system developer with a basis for their system development and integration i.e. the authors have not dwelled on the physical architecture of the developed architecture.
2.1.3 Hierarchical Framework for CBM

Wang et al [15] proposed a hierarchical framework for condition based maintenance as shown in Figure 4. A hierarchical architecture is also known as layered architecture. The
The proposed CBM system has a hierarchical structure for implementation of condition based maintenance. This structure is disadvantageous because lower layers in the hierarchy cannot communicate with the upper layers, for example, the prognostics in Figure 4 cannot be performed unless an abnormal state is detected in the system. Hence, this sort of hierarchical data flow leads to added latency time compared to parallel data flow. The databases shown in the Figure 4 serve as local repository of information which is used by other functional layers of the system or as a repository for historical data. The authors have not demonstrated a practical implementation of a system with this architecture. Hence the architecture has not been validated.

2.1.4 Open Architecture for Data Mining and Analysis

Gorinevsky et al [16] proposed an architecture as shown in Figure 5 for the anomaly detection. This architecture does not support prognostics or continuous monitoring. It is used for analyzing historically stored data from the systems and detects anomalies i.e. offline monitoring. Hence it cannot be used in real time monitoring. This resulting information can be remotely accessed using a web browser.

![Figure 5 Architecture for Data Mining and Analysis](image)

The test data were obtained by simulating Flight Operational data through NASA FLTz flight simulator. This data collect was monitored offline by the system to detect anomalies and how they affected the simulated flight.

2.1.5 Reconfigurable Prognostic Platform (RPP)

Liao et al [17] have proposed a prognostics framework called RPP for implementing prognostics in machine tools. The framework treats each prognostic algorithm as an independent component of the framework. The framework shown in the Figure 6 is composed of three agents namely the system agent (SA), knowledge database agent (KA) and executive agent (EA). The SA manages system resources; its functions include memory management and data acquisition. The EA contains various toolboxes as shown in the figure below. The KA provides decision making support. The EA along with SA and KA function as reconfiguration tools which help to reconfigure the system based on requirements.
The algorithm selection is based on a Quality Function Deployment method proposed by Yoji Akao in the 1960s. The tool essentially selects the algorithm based on application conditions such as signal characteristics (stationary or non-stationary), system knowledge (limited or enough), input data dimension etc. The authors also proposed an application and authentication server which serves as an interface between users and the database.

Figure 6 RPP Framework [17]

2.1.6 .NET Framework for Fault Diagnosis and Failure Prognosis

Chen et al. [18] proposed a new system architecture for fault diagnosis and failure prognosis based on .NET framework, as shown in Figure 7. Four constituent components in the system, including data processing, feature extraction, fault diagnosis and failure prognosis, are built as individual .NET components with associated graphical user interfaces that can be independently used.
The proposed .NET framework based architecture has the features of modularity, flexibility, interoperability, easy use, update and deployment. Also, a probabilistic approach that uses Bayesian estimation called particle filtering was integrated in the system to perform the diagnostics and prognostics in a statistical way. The proposed system architecture was tested in real-world applications, such as bearing spalling fault diagnosis and failure prognosis and brushless DC motor turn-to-turn winding fault diagnosis. The results demonstrated the effectiveness and efficiency of the system architecture. Also, nearly real time implementation of the system was achieved, which shows much faster running speed than that in the MATLAB.

2.1.7 E-Maintenance Conceptual Framework

E-Maintenance is an emerging maintenance support concept which implies maintenance with the aid of Information and Communication Technologies (ICT). The goal of e-maintenance is to provide a means of collaboration between corporate planning tools such as Enterprise Resources Planning (ERP) (a repository of all the management information integrating finance, accounting, manufacturing, sales and customer relations management within an organization) and the Computerized Maintenance Management System (CMMS) (a database of companies maintenance operation).
Significant work has been done in the field of e-maintenance for integration of corporate planning and resource management. Takata et al [22] have proposed a web based maintenance system. Hung et al [23] proposed a security framework for the e-maintenance for data integrity in a web based e-maintenance system. Macchi et al [24] did a study on information requirements for an e-maintenance system. This study demonstrated that equipment level data analysis may not be sufficient to maintenance decision making in manufacturing facility. Levrat et al [19] proposed a framework for e-maintenance which can take support from the ICT in the CBM system. This framework was essentially developed for assisting in maintenance decision making in manufacturing facilities. The framework was modeled based on Zachman framework for enterprise architecture which helps in integrating the business process with maintenance scheduling [25]. The authors proposed functional frameworks to support the e-maintenance functions in a manufacturing facility. Figure 8 illustrates a typical architecture for an e-maintenance system.

2.2 PHM Physical Architectures

Physical architecture describes the flow of information between various operational resources in the system architecture. They describe how these resources are used in the system and how they interact. The following are some of the physical architectures proposed by various researchers.
2.2.1 NASA’s Livingstone Architecture

Schwabacher et al [20] proposed a physical architecture for the X-37 orbital vehicle. The architecture for the system is as shown in the Figure 9. The objective of this system was real time processing of health data with diagnostic and prognostic capabilities. The sensor data feature extraction functionality is embedded in the Vehicle Management Software (VMS) on the Vehicle Management Computer (VMC).

![Figure 9 Livingstone Architecture [20]](image)

The livingstone performs the diagnostics and prognostics on the data. Some of the drawback of the architecture as cited by the authors included inability of this system to handle the processing loads of the algorithm thereby leading to system crashes. Another major limitation the authors cited was the limited study on the interfaces required for this system so as to collect data from the sensors and discharge the required data to the end users.

2.2.2 Distributed PHM Architecture

Saha et al [21] proposed a distributed PHM architecture in power system applications. The authors’ used Small Programmable Object Technology (SPOT) devices which essentially serve as the distributed computing and sensing elements (CE) in this architecture and are used for data acquisition as well as diagnostics and prognostics. The distributed computing elements (CE) communicate with the base station for any critical updates regarding its mode of operation (i.e. diagnostics or prognostics mode) and this base station serves as an interface between SPOT devices and the central server. The SPOT device is a battery powered microcontroller capable of wireless communication.
The diagnostic and prognostic subroutines utilize particle filter framework. The authors suggest that the load for the prognostic computation is shared by the independent computing elements because prognostics is computing intensive and require more processing power. The computing elements in the architecture run the diagnostic algorithm in default mode. In case anomalies are detected in the components, prognostics is performed for the components. The system architecture is as shown in Figure 10.

Figure 10 Distributed PHM Framework [21]

The authors have not addressed issues related to middleware architecture for these distributed computing elements. This middleware framework has support these distributed processing elements to interact with each other for condition monitoring and prognostics.

3. SUMMARY AND CONCLUSIONS

One of the aims of this paper was to identify proposed functional and physical architectures for a PHM. Various architectures proposed in the past are identified to that end. An architectural framework typically describes what all the components are in the system, how these components interact with each other or with external environment and what the resources are required for these components to perform their functionalities.

The Open System Architecture – Condition Based Maintenance specifies a set of interacting methods for communication between various components in the system. These methods are analogous to a common language for communication. But these interacting methods required a middleware for communication. The middleware is essentially a software service used by software components to interact with each other. Some examples of the middleware are Common Object Request Broker Architecture (COBRA), Distributed Component Object Model (DCOM), Java Remote Method Invocation (RMI) etc. But there are not sufficient studies on which middleware framework is appropriate. In the absence of middleware it becomes impossible to disseminate the information within a system.
The e-maintenance framework provides architecture for integration of various corporate resource management solutions with the PHM system using web technology for maintenance planning in a manufacturing facility. Hence, it essentially provides a framework for assisting in maintenance decision making in a manufacturing facility with the aid of information and communication technologies. The current literature on e-maintenance suggests it is predominantly a web based technology and it does not address the issue related to middleware architecture identified previously.

Choosing an appropriate algorithm for monitoring a particular system is a very vital and difficult task. The Reconfigurable Prognostic Platform (RPP) methodology proposed by Liao et al [17] using Quality Function Deployment (QFD) could be one such approach to solve the problem. But, the drawback of this approach is that the method takes the measured signal characteristics of the monitored system and the user requirements rather than taking into consideration any of the algorithm performance parameters which would give better prediction.

Architectures which have a high level of hierarchy have higher latency time in data transfer because the information needs to trickle through the higher layers to reach the lower layers in the hierarchy, such as the one proposed by Wang et al [15].

Without complementary functional and physical architecture the system cannot achieve its intended requirements. The livingstone architecture proposed by Schwabacher et al [20] is one such example where the functional blocks are not adequately supported by the computing infrastructure of the system. A distributed computing architecture can reduces the load on a single computing element by using distributed computing elements.

The .NET framework proposed by Chen et al requires a .NET framework for supporting the software. This framework is not available in open source operating systems such as Linux, and hence the software hence the software architecture of that sort is limited to windows based platforms.

4. REFERENCES
