An Extensible CBM Architecture for Naval Fleet Maintenance Using Open Standards

Sumant Tambe
Real-Time Innovations, Inc.
232, E. Java Dr. Sunnyvale, CA 94089
sumant@rti.com

Abdel-Moez E. Bayoumi, Alex Cao,
Rhea McCaslin, Travis Edwards
Condition-Based Maintenance Center
University of South Carolina, Columbia, SC 29208
bayoumi@sc.edu, caoa@sc.edu, mccaslir@email.sc.edu,
edwardt2@email.sc.edu

ABSTRACT

Condition-based maintenance (CBM) of naval assets is preferred over scheduled maintenance because CBM provides a window into the future of each asset's performance, and recommends/schedules service only when needed. In practice, the asset's condition indicators must be reduced, transmitted (off-ship), and mined using shore-based predictive analytics. Real-Time Innovations (RTI), Inc. in collaboration with the University of South Carolina CBM Center is developing a comprehensive, multi-disciplinary technology platform for advanced predictive analytics for the Navy's mechanical, electrical, and IT assets on-board ships. RTI is developing an open, extensible, data-centric bus architecture to integrate shipboard asset monitoring data with shore-based predictive analysis tools. The interoperability challenge is addressed using the Model-Driven Architecture (MDA) by transforming sensor data to rigorously specified standard data models. Our MDA process includes open standards such as the OMG Data Distribution Service (DDS) and Open System Architecture for Condition-Based Maintenance (OSA-CBM), both of which have enjoyed success in the Navy. Furthermore, the Navy's Information Assurance (IA) requirements are implemented using the OMG Secure-DDS standard. In summary, the technology will improve combat readiness using a truly interoperable data-bus for exchanging CBM data from ship-to-shore while reducing distractions to the sailors, standby inventory requirements, and decision time for analysts.

Keywords Condition-Based Maintenance (CBM), Data Distribution Service (DDS), Open System Architecture for Condition-Based Maintenance (OSA-CBM)

1. INTRODUCTION

Mission readiness and longevity of the Navy fleet depends heavily on how well-maintained its constituent systems are. Maintenance comprises a major portion of total ownership costs for Navy systems. Unnecessary maintenance contributes to inflated ownership costs and reduced readiness of deployable assets.

Condition-based maintenance (CBM) is a well-known predictive maintenance technique used in a number of industrial-scale engineering disciplines including mechanical, civil, chemical, and electrical, etc. A CBM approach helps identify components that are most likely to exhibit faults that require repair or replacement. Conversely, it also helps identify which equipment components remain in functional condition without the need for maintenance.

The US Navy and Department of Defense (DoD), in general, have embraced [2] CBM due to its ability to diagnose problems before they occur, improve overall mission reliability, reduce unnecessary downtime, and reduce inventory and unnecessary maintenance procedure costs. CBM differs from scheduled or preventive maintenance because CBM reduces the possibility of unplanned downtime when applied properly. Thus, CBM implies improved mission readiness and reliability at a reduced cost for the Navy.

CBM necessarily depends on (1) instrumentation of equipment, (2) continuous monitoring and collection of an asset's condition indicators, and (3) predictive analytics on the collected data. A key step is to

continuously monitor the health of a system by collecting data from sensors, analyzing it using artificial intelligence techniques, and identifying any anomalies and/or incipient failures to the stakeholders. In a Navy ship, the major subsystems include navigation, electrical, mechanical, communications, engine, propulsion, and the payload. Using the myriad variety of onboard sensors, it is now possible to estimate the overall health of the ship.

Developing and maintaining CBM systems have become more expensive with the increasing sophistication of diagnostics and prognostics. With shrinking defense budgets and the drive to improve acquisition flexibility, different subsystems in ships are built by different vendors and are assembled by integrators. The COTS components must interoperate with each other as well as the overall CBM system for effective prognostics. The architecture must not couple the sensors with the CBM system because more than one type of CBM system could be operational concurrently and the sensors and/or the CBM system may evolve independently as the providers of those systems may be different. When the sensors and/or the CBM system do actually evolve, it is imperative to ensure that existing CBM dataflows are not disturbed and that the new capabilities are seamlessly integrated into the overall system.

Another challenge in achieving full-fledged CBM capabilities in the US Navy is that the shipboard data is often isolated [3] from the shore-based analysis tools. Integrating the shipboard data silos with the predictive analytics tools on shore is challenging for several reasons:

1. **Incompatible Data Formats:** The data collected by the shipboard sensors must be available to the analysis tools in a form they understand. Common Information Model (CIM) [4] is a general-purpose specification to collect information about devices. CIM is an XML-based model. The analysis tools, however, may or may not understand CIM. They may not even use XML as data input.
2. **Disconnected, Intermittent, and Limited Connectivity:** The communication channel between the ships and the shore is often limited in capabilities. For instance, the connection may not be available all the time, the bandwidth may be scarce and opportunistic. The monitoring data must often compete with high-priority mission-related data streams. Any solution connecting the shipboard data silos to the shore-based analysis tools must address the connectivity problem.
3. **Data Volume:** Shipboard sensors gather large amount of instantaneous performance and device condition data. The predictive insights depend on trends—values that change over time as opposed to the instantaneous values. Indeed, the statuses of healthy devices/services do not change until they start to degrade. Therefore, transporting raw data to the shore through contested links is quite inefficient and counterproductive. The data may be analyzed, reduced, and prioritized before it is sent off-ship to optimize the usage of available link capacity. The connectivity solution must address the data volume challenge through smart data reduction/compression techniques.
4. **Information Assurance:** The ship-to-shore data transport solution must meet the Navy's information assurance (IA) requirements. For instance, not all performance data is relevant to all Subject Matter Experts (SME) involved in the process of performing predictive analytics. The data integration solution must be aware of the Navy's access control policies and must enforce them during ship-to-shore transmission as well as later during analysis.

In this paper, we present a novel extensible CBM architecture for Navy fleet maintenance. Our architecture is based on open standards that are already deployed in the Navy including the Data Distribution Service (DDS) and the Open System Architecture for Condition-Based Maintenance (OSA-CBM).

Our architecture is designed for a full-fledged CBM system and it is necessarily "formal". The data models and protocols in our architecture are open, formal (i.e., rigorously defined) and also standardized so that participant applications can be swapped in/out conveniently as long as they rely on the open architecture.

In our architecture, the innovation lies in (1) the analog sensors for condition detection, (2) distribution of data from acquisition to prognostics using standard data models and communication protocols, and (3) the analysis applications that run the CBM algorithms to generate alerts and maintenance recommendations. All protocols and

data formats are open. We show how DDS and OSA-CBM, which are well-established standards in their respective domains, are highly compatible together.

Next, we provide a brief overview of CBM and related standards and technologies.

2. CONDITION-BASED MAINTENANCE

Condition-based maintenance (CBM) is a transition from a reactive to a proactive maintenance approach. Maintenance actions are performed based only upon evidence of need. This differs from traditional management practices such as failure-based (corrective maintenance) or time-based (preventive maintenance) approaches. Implemented in the field, CBM employs component monitoring equipment to detect signs of wear and enable targeted maintenance.

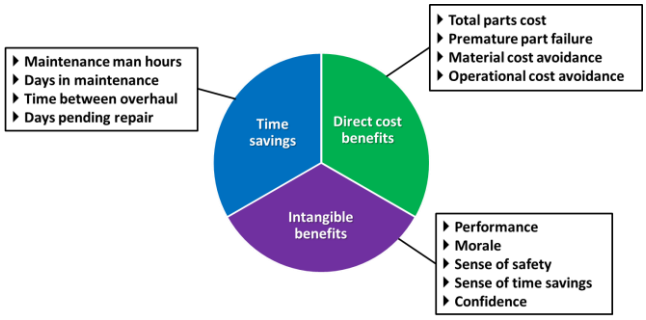


Figure 1: Benefits of CBM implementation

When implemented successfully, CBM can lead to an improvement in performance, productivity, and overall effectiveness of systems. The benefits of CBM can be realized through time savings, direct cost benefits, and/or intangible benefits (Figure 1). Time savings can be achieved through a reduction in maintenance man hours, time between failure, and the time an asset is not operational due to maintenance. Direct cost benefits are achieved by a reduction of costs in replacing parts and an increase in cost avoidance (material and operational). Time savings and direct cost benefits are both quantitative tangible metrics that can be measured in dollars or time. Intangible benefits are qualitative and include an improvement in sense of safety and an increased feeling of confidence and morale among users which lead to better personnel performance. Another benefit that CBM provides over traditional maintenance practices is a greater focus on and understanding of problems that occur in between scheduled maintenance, which can lead to better initial design of components.

CBM can be applied to many areas including, but not limited to aviation, transportation, energy, and civil structures. Any field with complex mechanical systems and high operational readiness requirements is a good candidate for CBM. The goal of CBM is to reduce the maintenance burden on the sailor, maintain or enhance safety, extend time between overhaul (TBO), increase availability and readiness of assets, and reduce operating and support costs.

CBM at the University of South Carolina encompasses a wide variety of areas including testing, natural language processing, diagnostics and prognostics, signal processing, cost-benefit analysis, tribology, and modeling and simulation.

2.1 The OSA-CBM Specification

The OSA-CBM specification [10] is an open standard architecture for moving information in a condition-based maintenance system. OSA-CBM was developed in 2001 by an industry led team partially funded by the Navy through a Dual Use Science and Technology (DUST) program. OSA-CBM is now managed and published by the Machine Information Management Open Systems Alliance (MIMOSA) standards body. Its goal is to enhance interoperability between multiple vendors' software components [11].

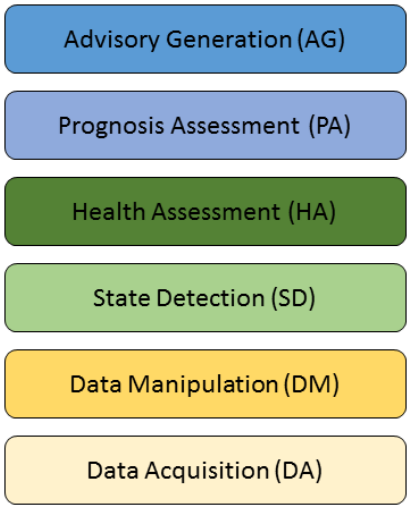


Figure 2: OSA-CBM Functional Blocks

This section provides a short overview of the OSA-CBM specification. A much more comprehensive overview of the OSA-CBM specification is presented in [12][13].

The OSA-CBM architecture is divided into the interface specification and functional blocks (Figure 2). These specifications are defined using the Unified Modeling Language (UML) and intended to be

platform independent which can be mapped into various programming languages and middleware technologies. Vendors and integrators can implement the standard using the appropriate technology for their environment. For example, while aircraft health management vendors may elect to use a "real-time" implementation, a vendor developing a portable maintenance aid may elect to implement the standard using XML and web services.

OSA-CBM specifies a standard architecture and framework for implementing condition-based maintenance systems. It describes the six functional blocks of CBM systems, as well as the interfaces between those blocks. The standard provides a means to integrate many disparate components and eases the process by specifying the inputs and outputs between the components. In short, it describes a standardized information delivery system for condition-based monitoring. It describes the information that is moved around and how to move it. It also has built in metadata to describe the processing that is occurring.

The OSA-CBM data model is based on the concept and employment of metadata (data about data), i.e. OSA-CBM data are always identifiable and traceable. The aim is to have data that supports data-centric maintenance information management. In fact, OSA-CBM data can be directly mapped into any OSA-EAI-compliant relational database maintenance systems with ease.

There are four primary OSA-CBM data classes: DataEvent, Configuration, Explanation and Extensible. DataEvent is the dynamic data related to condition monitoring events generated by an OSA-CBM module such as measurements, manipulated or processed data, etc. The DataEvent class forms a substantial part of the OSA-CBM data model and our prototype implementation of OSA-CBM uses DataEvent and related classes.

2.2 USC's Experience with the South Carolina Army National Guard

Since 1998, the University of South Carolina has collaborated with the South Carolina Army National Guard (SCARNG) on projects involving the implementation of CBM in Army rotorcraft fleets. Specifically, these projects worked with the AH-64, UH-60, and CH-47, and were aimed at reducing the Army aviation costs and increasing operational readiness [5][6][7]. CBM implemented aircraft have shown an increase in operational readiness resulting in millions of dollars being saved in operational costs.

One use case study was the cost-benefit analysis of an on-board vibration monitoring system known as the Vibration Management Enhancement Program (VMEP). The objective was to provide annual cost savings analysis of VMEP implementation and create a cost-benefit analysis model. The analysis showed a decrease in operational costs and an increase in mission capability through a reduction in unscheduled maintenance for VMEP-equipped aircraft versus non-VMEP equipped aircraft [8].

Another case study involved the testing of the AH-64 tail rotor gearbox. The aircraft model and component were chosen based on an analysis of historical data sources that showed a high number of incidences involving the gearbox and leaking liquid. The research at USC led to changes in maintenance practices in the field along with a reduction in part costs, unscheduled downtime, and maintenance man hours [9]. A technical bulletin was released followed by an IETM update because of the change in maintenance practice. Tangible benefits of the study include $2.98M/yr in cost avoidance due to reduced part demands. This project led to the highest ROI-to-date (20.2:1) in U.S. Army aviation history.

3. PREVENTIVE PREDICTIVE MAINTENANCE (PPM) APPROACH

The University of South Carolina is developing an integrated approach that encapsulates both data-driven and physics-based modeling techniques in order to build accurate diagnostic and prognostic models. This approach, shown in Figure 3, is called Preventive Predictive Maintenance (PPM) and can be divided into three phases:

1. Phase I – Measurement
2. Phase II – Physics Modeling
3. Phase III – Integration & Validation

Phase I and Phase II can be implemented independently and each have their own advantages and disadvantages.

3.1 Measurement

Phase I can be described as an integrated data-driven approach based on three distinct data sources: on-board sensor data, historical asset data, and test data.

3.1.1 On-Board Sensor Data

Health and Usage Monitoring Systems (HUMS) collect and process on-board sensor data using signal processing techniques to extract features to calculate condition indicators (CIs). These CIs are then used as inputs to the analytical model. Advanced signal processing techniques such as higher order spectral analysis and joint time-frequency analysis are employed to extract additional non-linear features from the sensor signals that traditional techniques cannot.

3.1.2 Historical Asset Data

Aircraft data is collected from a variety of sources (e.g., maintenance records, operator logs, parts inventory, etc.). These data are collected for different purposes and reported in different formats. As is, these qualitative disparate data sources cannot be combined together with quantitative data (on-board sensor data) in analytical models. Tools such as natural language processing are used to convert any unstructured text data (e.g., maintainer's notes) into concepts and types that can be categorized and analyzed. This will transform the data into a standardized machine-analyzable form. Data then can be correlated, integrated, and categorized into a form that fits the input of an analytical model.

3.1.3 Testing Data

Testing is used to characterize or refine diagnostic/prognostic algorithms and to understand the physics or root causes of faults. The testing of faulted versus unfaulted components allows the definition of boundaries separating normal operating condition with abnormal conditions. Moreover, fault testing allows validation and refinement of existing diagnostic/prognostic algorithms. More specifically, seeded fault testing allows the examination of possible fault progression and severity in a controlled reproducible environment. Test data from fault testing and failure modes, along with the qualitative data gained from inspections or tear down analysis is used to correlate prognostic algorithms with faults and degree of severity. This enhances understanding of the underlying problem related to physical parameters (e.g., gear failure). This iterative process produces fault-correlated algorithms based on historical knowledge and failure modes.

3.1.4 Integrated Dataset

These three distinct datasets allow comprehensive predictive models (classification, segmentation, association, etc.) to be created from statistical and machine learning techniques. Health indicators can then be created from built diagnostic models. The integrated dataset would be a comprehensive description of the asset and contain both normal and abnormal operating conditions. Currently, most condition-based maintenance implementations focus solely on sensor data and do not consider other sources of data like logistics. It is believed that logistics data contains useful information that has not yet been used for modelling purposes. In essence, latent relationships exist between on-board sensor data and logistics data that can be uncovered in an integrated dataset.

3.2 Physics Modeling

Phase II is a physics-based approach. In order to monitor the health of components and detect failures, an understanding of the physics of the failure mechanisms is needed. This requires the correlation of system data with component usage. Historically, these correlations were developed based on costly instrumentation, physical tests and empirical relationships. A theoretical framework based on component, subsystem, and integrated system models can be used to effectively understand the physics of failure modes. This knowledge will help to determine root causes of failures and the optimal locations for sensor placement.

The data from the model's responses from applied inputs is used for the development of prognostic algorithms that are sensitive to specific faults and failure modes. Model inputs can be velocity, forces, heat source, cracks etc. Model response would be in terms of stress, strain, temperature, etc. These models would encapsulate the dynamic, acoustic, material (stress, strain), and thermal response of component and subsystem models and relate them to the overall system model. The advantage of this approach is that it allows the emulation of faults and failure mechanisms that are not practical to create in a field or test environment.

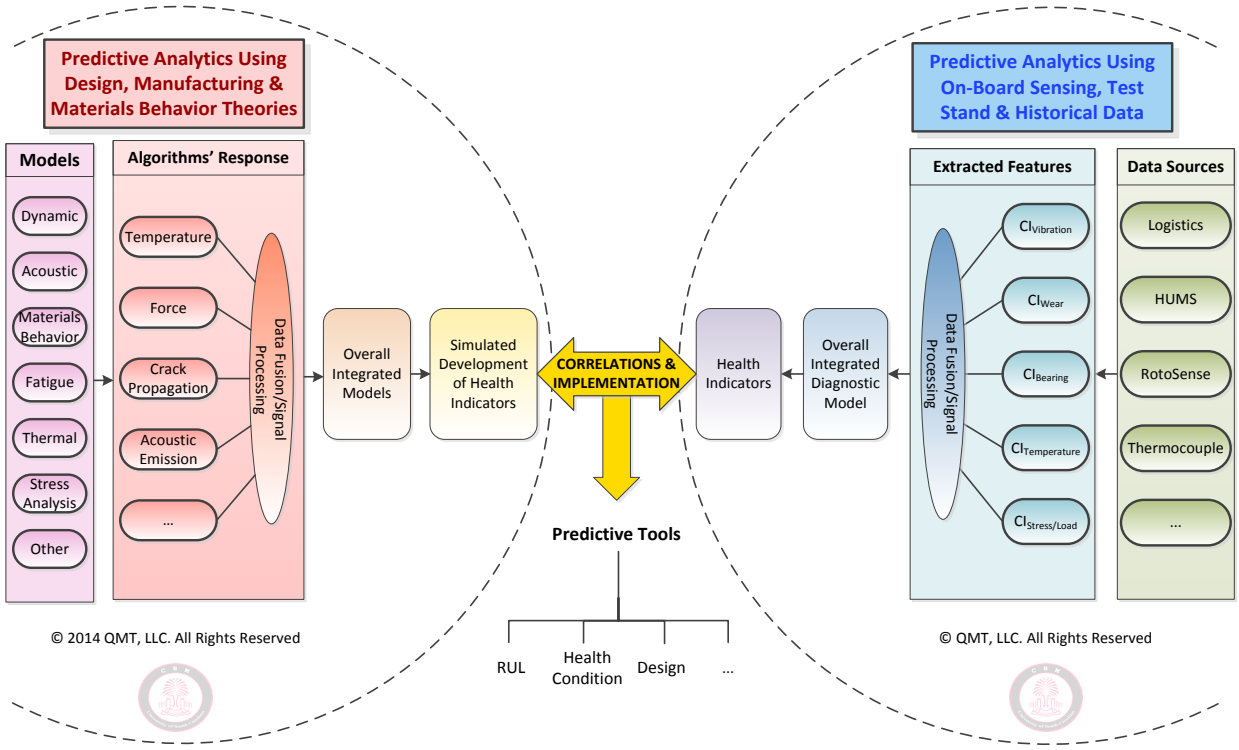


Figure 3: University of South Carolina's Preventive Predictive Maintenance Approach

In addition, data fusion of different simulation models would take into account potential model interactions yielding a richer source of information than a single model by itself. An integrated model can ascertain the effect that a fault has on the functionality of the various elements of the system. More importantly, it allows us to correlate theoretical analysis with testing data allowing us to refine the numerical model and modify the test setup.

3.3 Integration & Validation

Phase III involves the correlation and integration of these two approaches. The integrated system model will greatly enhance the development of predictive tools while also helping to support and inform the historical and test data collected. Simulation data supplements historical aircraft data not feasibly obtainable and allows exploration of various failure modes and root causes. Conversely, the reliability and robustness of the modeling and simulation capability of the proposed algorithm can be demonstrated and correlated with sensor data from historical or test data. The integration of the two parts allows for predictive tools to be created for remaining useful life, health condition, and component design. This is a direct outcome of the PPM approach. The intersection of design, models and simulations, and real world systems is where new tools are being developed to predict health, life, and performance of components, subsystems, and complete systems.

4. Overview of Data Distribution Service

Data Distribution Service (DDS) is the first open international middleware standard addressing publish-subscribe communications for distributed, real-time and embedded systems. The DDS API and several other satellite technical specifications, such as DDS Wire Interoperability specification, Extensible and Dynamic Topic Types (X-Types) specification, modern language bindings for C++ and Java are standardized by the Object Management Group (OMG)—an internationally recognized standards body most widely known for UML and CORBA.

The core DDS specification defines a data-centric publish-subscribe architecture (see Figure 4) for connecting anonymous information providers with information consumers. DDS promotes loose coupling between system components. The information consumers and providers are decoupled with respect to time (i.e., they may not be present at the same time), space (i.e., they may be anywhere), flow (i.e., information providers must offer equivalent or better quality-of-service (QoS) than required by the consumers), behavior (i.e., business logic independent), platforms, and programming languages.

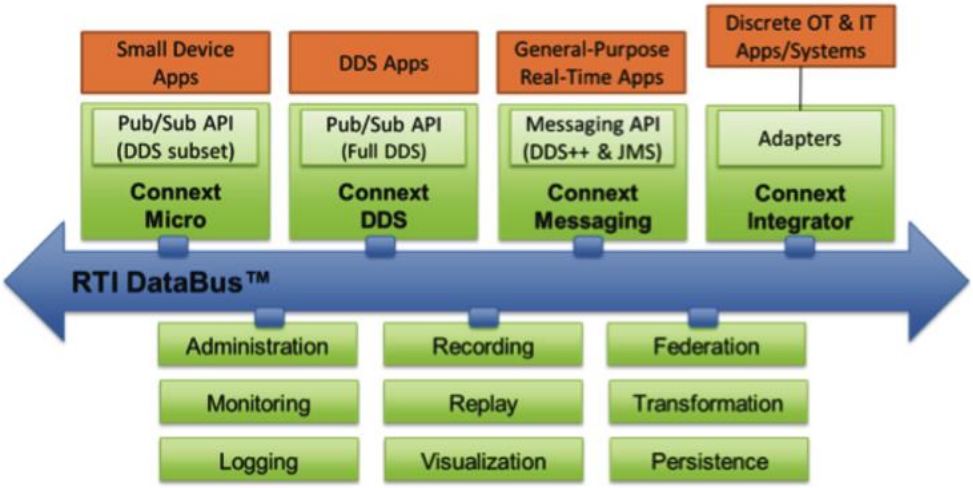


Figure 4: DDS Data Bus

A data provider publishes typed data-flows, identified by names called "topics". The coupling is expressed only in terms of topic name, datatype schema, and the required and offered QoS attributes of consumers and producers respectively. DDS allows fine control over data delivery by means of standard QoS policies, such as durability, reliability, history, deadline, time-based filtering, liveliness, transport priority, resource limits, and more.

Below we provide an overview of select DDS concepts used in our approach.

1. **Data-Centric Architecture:** DDS facilitates data-centric architecture where applications share a global data space governed by schemas specified using the XTypes2 standard. Each point-to-point dataflow is described using a structured datatype (e.g., an Interface Definition Language struct). The datatype could be keyed on one or more fields. Each key identifies an instance (similar to a primary key in a database table) and DDS provides mechanisms to control the lifecycle of instances. Instance lifecycle supports CRUD (create, read, update, delete) operations. Complex delivery models can be associated with dataflows by simply configuring the topic QoS.
2. **DataWriter and DataReader:** DataWriter and DataReader are end-points applications use to write and read typed data messages (samples) from the global data space. DDS ensures that the end-points are compatible with respect to the topic name, datatype, and the QoS. Creating a DataReader with a known topic and datatype implicitly creates a subscription, which may or may not match with a DataWriter depending upon the QoS.
3. **Data Caching:** DDS is not just a messaging middleware, although it can be configured to behave like that. DDS, DataReader in particular, provides caching of samples and different APIs to traverse the cached data samples so that applications need not make copies. DDS distinguishes between read and take, where read keeps the data in middleware cache until it is removed by either calling take or overwritten by subsequent samples. Resource limits QoS prevents middleware caches from growing out of bounds. The DataReader cache can be queried using specific instances as well as iterate over all the instances observed by the system. Finally, query conditions provide a powerful mechanism to write SQL-like expression on the datatype members and retrieve samples that satisfy the predicate.
4. **Content-Filtered Topics:** It specifies a refined subscription that filters samples that do not match an application-specified predicate. The predicate is a string encoded SQL-like expression based on the fields of the datatype. The query expression and the parameters may change dynamically. Filtering of samples could take place before publication or upon reception.
5. **DDS Quality-of-Service:** The DDS standard supports 22 different QoS policies to control various aspects of data delivery including reliability, persistence, deadline, resource usage, fault-tolerance, etc. We describe the significance of QoS policies used in our solution below:

6. **Reliability:** It controls the reliability of the dataflow between each pair of DataWriters and DataReaders. BEST_EFFORT and RELIABLE are two possible alternatives. BEST_EFFORT reliability minimizes cpu/memory resource consumption. RELIABLE, on the other hand, uses an ack/nack based protocol to provide a spectrum of reliability guarantees from strict (fully reliable) to BEST_EFFORT. The reliability can be tuned using the History QoS policy.

5. Integrating DDS and PPM in an Extensible CBM Architecture

Figure 5 shows our proposed architecture for combining PPM and DDS to develop an extensible architecture for Navy CBM applications. Data acquisition and health assessment applications are OSA-CBM compliant and they communicate using DDS instead of other integration technologies, such as Web Services or relational databases. Of course, the communication between the applications is data-centric, thanks to DDS.

The key benefit of using DDS is that the application integration problem is substantially simplified due to the bus architecture. A DDS application requires only one additional connection to the global data space to enable communication with all the existing applications. The middleware ensures that the applications that use compatible topic name, topic type, and QoS discover each other automatically and begin exchanging data. Point-to-point integration technologies such as Simple Object Access Protocol (SOAP) and sockets on the other hand would require one connection for each existing application that it exchanges data with. Therefore, to communicate with N applications, N point-to-point links are necessary.

Furthermore, point-to-point communication makes independent evolution of applications extremely difficult. When the data provider application evolves, it cannot communicate with older versions of consumer applications unless the data model differences are explicitly addressed at application-level. With each new version of the application the problem of data model differences exacerbates, which increases development, testing costs, and time to market.

5.1 Modeling OSA-CBM Data in DDS

In our architecture, the producer and consumer applications continue to use the OSA-CBM compliant XML data format for sharing information. The global data space captures the OSA-CBM data model using DDS X-Types [14]. Having the model captured in X-Types allows the model and/or the applications to evolve without disrupting the applications that do not evolve.

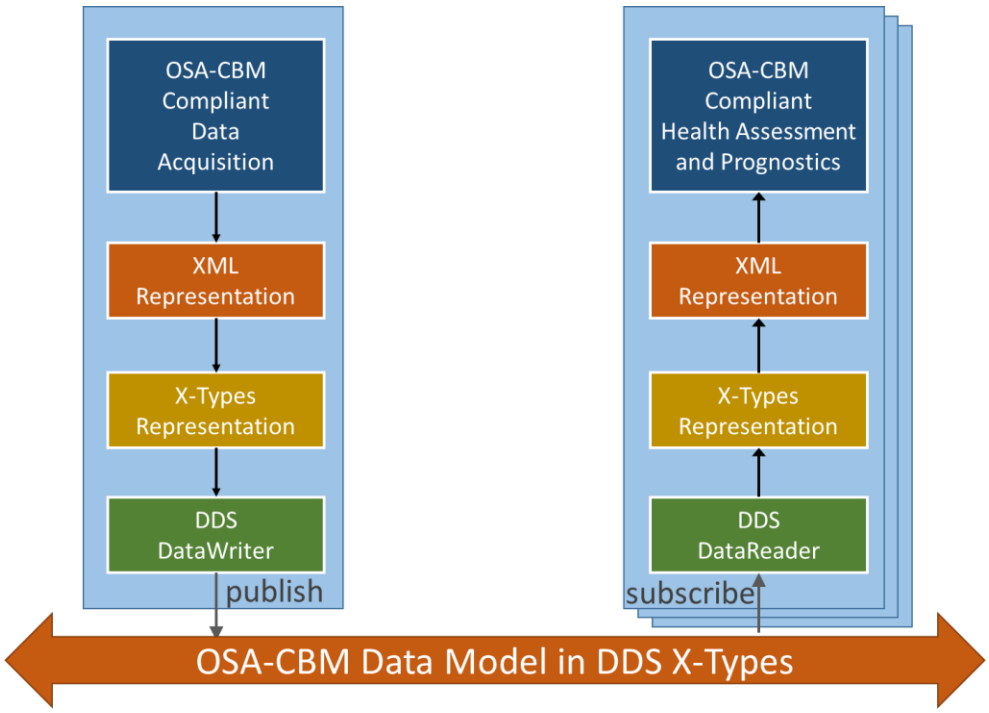


Figure 5: Data Bus Architecture for Condition-Based Maintenance using DDS and OSA-CBM

For efficiency, XML-based CBM data must be converted into the data representation used by the global data space, which is binary. The conversion is achieved using an adapter that converts the OSA-CBM compliant XML data into its equivalent X-Types data representation (and vice versa). The adaptation is semantically equivalent because both data representations are obtained from the common OSA-CBM data model.

There are two possible approaches to implement OSA-CBM to X-Types transformation.

5.1.1 Gateway Approach

A gateway is an adapter process (perhaps implemented using the RTI Routing Service), which provides a pluggable framework to develop protocol and data-format bridges. A pair of gateway instances would mediate the communication between two applications and transform the incoming XML messages over say, TCP protocol, to equivalent DDS-XTypes samples over UDP, which is the most commonly used communication protocol in DDS deployments. A reverse transformation is performed at the receiving side to convert DDS samples back to XML.

The key thing to note here in this approach is the applications are oblivious to the intermediate transformation. No changes are necessary to the OSA-CBM compliant sensors and/or applications. The OSA-CBM compliant applications are oblivious to the fact that they are communicating over DDS while they reap the benefits of it.

5.1.2 Code Generation Approach

Second approach uses direct mapping of the OSA-CBM types to DDS-XTypes using some sort of code generation. MIMOSA bundles standardized eXtensible Schema Definitions (XSD) for the OSA-CBM specification. The set of XSD files describe hundreds of types that encapsulate information at various functional blocks of the OSA-CBM specification. A code generator would accept the XSD files as input and would produce C/C++/Java code that captures the OSA-CBM types using the idioms of the target language.

Even though the second approach is theoretically straightforward, there are practical limitations. As of this writing, we are not aware of any DDS implementation that supports the OSA-CBM XSD out-of-the-box. We have, therefore, invented a novel approach to map the OSA-CBM XSD to DDS-XTypes indirectly, which we describe next.

5.2 RefleX: Reflection-based Type Modeling for DDS-XTypes

RefleX [17] is short for Reflection for DDS-XTypes. The crux of this C++ library is to create DDS-XTypes compliant type representations directly from native C++ types. RefleX is declarative (i.e., it is not a reflection API). There is no separate code generation step involved (other than compilation). The RefleX library accepts application-level datatypes, no matter how complex, and maps them to equivalent DDS topic types.

The application-level datatypes are generated using COTS XML data-binding tools. (e.g., Code Synthesis XSD [15], JAXB [16]). The benefit of using the existing COTS code generation tools is that significant efforts have been put into developing robust code generators that are compliant to the XSD standard. As a consequence, OSA-CBM XSD is handled extremely well.

As RefleX is a C++ library, our evaluation is limited to the C++ language only. A similar approach is also conceivable in Java, which natively support reflection capabilities. C++, on the other hand, does not support reflection and therefore, programmers must provide some additional information declaratively so that the RefleX library can transform native C++ structured types to DDS-XTypes representation. We developed the `RTI_ADAPT_STRUCT` macro to simplify the task.

Listing 1 shows a code snippet that adapts `osacbm::DataEvent`, `osacbm::DADataEvent`, and `osacbm::DADataSeq` C++ classes to DDS-XTypes. The classes are related by inheritance relationship as shown on the right in Listing 1. The classes are mapped to equivalent `struct` and `valuetypes` in DDS-XTypes at program compilation time. The `RTI_ADAPT_STRUCT` macro does not affect the original source code of the classes. The macro is only *additive*.

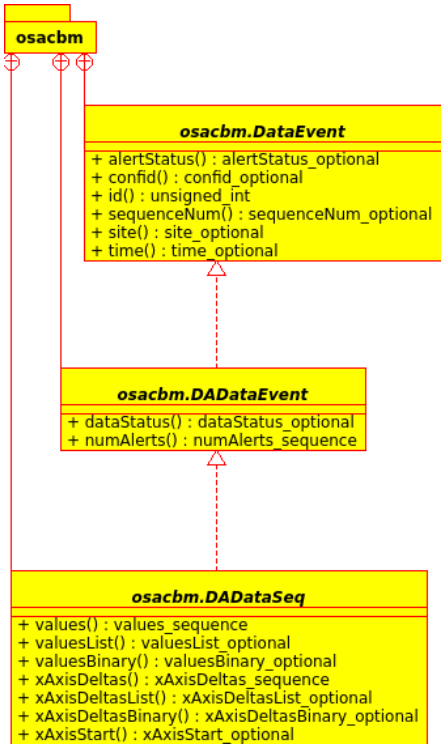
As long as the above code snippet is available during program compilation, DDS `DataWriter` and `DataReader` can be used directly with the aforementioned types and instances thereof. No IDL or separate code generation step is necessary.

The source code for the RefleX library can be downloaded from [17].

```
RTI_ADAPT_STRUCT(
  osacbm::DataEvent,
 (osacbm::DataEvent::alertStatus_optional,  alertStatus())
 (osacbm::DataEvent::confid_optional,       confid())
 (unsigned_int,                             id())
 (osacbm::DataEvent::sequenceNum_optional,  sequenceNum())
 (osacbm::DataEvent::site_optional,         site())
 (osacbm::DataEvent::time_optional,         time())
);

RTI_ADAPT_VALUETYPE(
  osacbm::DADataEvent,
  osacbm::DataEvent,
 (osacbm::DADataEvent::dataStatus_optional, dataStatus())
 (osacbm::DADataEvent::numAlerts_sequence,  numAlerts())
);

RTI_ADAPT_VALUETYPE(
  osacbm::DADataSeq,
  osacbm::DADataEvent,
 (osacbm::DADataSeq::values_sequence,            values())
 (osacbm::DADataSeq::valuesList_optional,        valuesList())
 (osacbm::DADataSeq::valuesBinary_optional,      valuesBinary())
 (osacbm::DADataSeq::xAxisDeltas_sequence,       xAxisDeltas())
 (osacbm::DADataSeq::xAxisDeltasList_optional,   xAxisDeltasList())
 (osacbm::DADataSeq::xAxisDeltasBinary_optional, xAxisDeltasBinary())
 (osacbm::DADataSeq::xAxisStart_optional,        xAxisStart())
);
```



Listing 1: Left: Adaptation of Select OSA-CBM Types to DDS-XTypes. Right: Inheritance Relationship

6. RELATED WORK

Sreenuch et al. [12][13] describe an application of data distribution service for implementing the Open System Architecture Condition-Based Maintenance (OSA-CBM) standard for Integrated Vehicle Health Management (IVHM). The authors argue that the primary benefit of using the publish/subscribe architecture is to abstract the message transport from the software components and allows them to be decoupled. Furthermore, deployments can be done rapidly. Our research not only support this prior research but also goes a step further by demonstrating the use of DDS-XTypes to create an extensible data model and distributed architecture for CBM. The novelty of our approach lies in extensibility and the normalized data model for sharing CBM data.

Software Health Management (SHM) [18] is an emerging discipline that applies the principles and techniques of system health management and maintenance to software systems. SHM is a promising approach to providing fault-tolerance in real-time systems because it not only provides for fault detection and recovery but also effective means for fault diagnostics and reasoning, which can help make effective and predictable fault mitigation and recovery decisions.

Towards that end, previous work [19] outlined a software intensive approach for avionics sensor health management using probabilistic belief networks. We leveraged our extensible DDS data-bus architecture not unlike the one presented in this paper. The key difference being that in [19] we used a custom data-model for data acquisition and distribution of health assessment results. This paper uses the standard OSA-CBM data model.

Our future work includes exploring novel middleware adaptation techniques, such as GRAFT [20] and SafeMAT [21] for distributed resource monitoring, safe failure isolation, adaptive failover decisions, and

subsystem diagnostics in an end-to-end CBM solution.

7. CONCLUSION

Condition-based maintenance (CBM) is a transition from a reactive to a proactive maintenance approach and has shown its effectiveness in a number of engineering domains. University of South Carolina CBM Center has developed and successfully demonstrated a number of advanced CBM techniques over the last 15 years. Implementing these CBM techniques for fleet maintenance in Navy must overcome additional challenges due to limited connectivity, data volume, and incompatibilities in data sharing.

We described an extensible standards-based solution to address the key architectural challenges. Our solution relies on DDS and OSA-CBM standards, both of which have enjoyed success in the Navy. We presented two approaches to implement the OSA-CBM data model using DDS. We discussed practical techniques to combine truly interdisciplinary research and reap the benefits for the Navy.

8. AUTHOR BIO

Sumant Tambe, Ph.D. is a Principal Research Engineer at Real-Time Innovations, Inc. Dr. Tambe's research interests include distributed real-time embedded systems, model-driven engineering, multi-paradigm software design, functional reactive programming, and modern C++. He has served as Principal Investigator on multiple SBIR research projects funded by a number of government agencies.

Abdel-Moez E. Bayoumi, Ph.D., is a Professor, an Associate Dean, and the Director of the Condition-Based Maintenance Center at the University of South Carolina. He has published 3 book chapters, over 100 papers and has over 17 years of experience in CBM. He received his Ph.D. degree from North Carolina State University in 1982.

Alex Cao is a Research Engineer with the Condition-Based Maintenance Center at the University of South Carolina. He has been working with the team for 3 years and is working with sensors, algorithms, data analytics, and high performance computing applied to CBM. He received his Master's degree from the University of Michigan in 2002.

Rhea McCaslin is a Graduate Research Assistant at the University of South Carolina with the Condition-Based Maintenance Center. She has been with the team for three years and is working on natural language processing and data analytics of CBM data. She will receive her Master's degree in Computer Engineering from the University of South Carolina in 2015.

Travis Edwards is a Graduate Research Assistant at the University of South Carolina with the Condition-Based Maintenance Center. He has been with the team for over three years and currently manages multiple projects dealing with the AH-64 and the sustainability of the rotorcraft fleet. He will receive his Master's degree in Mechanical Engineering from the University of South Carolina in 2015.

9. REFERENCES

[1] Rowden, Tom. "Littoral Combat Ship: All Head Full!" Proceedings Magazine. January 2013.: Vol. 139/1/1, 319.

[2] Steven W. Butcher. "Assessment of Condition-Based Maintenance in the Department of Defense": LG903B1, August 2000

[3] Navy SBIR 2014.1 Solicitation - Topic N141-030 "Sense and Respond Technology Enabling Condition Based Maintenance (CBM)" http://www.navysbir.com/n14_1/N141-030.htm

[4] Distributed Management Task Force Common Information Model. http://www.dmtf.org/standards/cim

[5] Giurgiutiu, V., Grant, L., Grabill, P., Wroblewski, D., "Helicopter Health Monitoring and Failure Prevention through Vibration Monitoring Enhancement Program", International Journal of Condition Monitoring and Diagnostic Engineering Management, UK, 4(4), 2001, pp. 33-40.

[6] Bayoumi, A., Eisner, L., "Transforming the US Army through the Implementation of Condition Based Maintenance", Journal of Army Aviation, May 2007.

[7] Bayoumi, A., Goodman, N., Shah, R., Blechertas, V., "Mechanical Diagnosis and Prognosis of Military Aircraft: Integration of Wear, Vibration Time-Frequency Analysis and Temperature into Diagnosis Algorithms", Proc. Of Advanced Materials For Application In Acoustics And Vibration, Cairo, Egypt, 2009.

[8] Bayoumi, A., Ranson, W., Eisner, L., Grant, L.E., "Cost and effectiveness analysis of the AH-64 and UH-60 on-board vibrations monitoring

system", Aerospace Conference, 2005 IEEE, 5(12), 2005, pp 3921-3940.

[9] Goodman, N., Bayoumi, A., Blechertas, V., Shah, R., Shin, Y., "CBM Component Testing at the University of South Carolina: AH-64 Tail Rotor Gearbox Studies," Proceedings of AHS International Specialists' Meeting on Condition Based Maintenance, Huntsville, AL, 2009.

[10] MIMOSA, OSA-CBM UML Specification 3.3.1 Release, Machine Information Management Open System Alliance, Tuscaloosa, AI, 2010

[11] M. Tiemann, An objective definition of open standards, Computer Standards & Interfaces 28 (5) (2006) 495–507.

[12] T. Sreenuch, A. Tsourdos, I. K. Jennions, "Distributed embedded condition monitoring systems based on OSA-CBM standard", Elsevier Journal in Computer Standards & Interfaces Volume 35 Issue 2, February, 2013, Pages 238-246

[13] Tarapong Sreenuch, Antonios Tsourdos, Ian Jennions, and Peter Silson, "Application of the Data Distribution Service for Implementing OSA-CBM Standard", Infotech@Aerospace 2011.

[14] Extensible and Dynamic Topic Types, OMG document formal/2012-11-10, http://www.omg.org/spec/DDS-XTypes/1.0/

[15] Code Synthesis XSD, www.codesynthesis.com/products/xsd/

[16] Java Architecture for XML Binding (JAXB), http://www.oracle.com/technetwork/articles/javase/index-140168.html

[17] RTI RefleX Library, http://rticommunity.github.io/rticonnextdds-reflex/

[18] A. Srivastava and J. Schumann, "The case for software health management," in Space Mission Challenges for Information Technology (SMCIT), 2011 IEEE Fourth International Conference on. IEEE, 2011, pp. 3–9.

[19] Sumant Tambe, Fernando Garcia Aranda, Joe Schlesselman, "An Extensible Architecture for Avionics Sensor Health Assessment Using Data Distribution Service", In the Proceedings of the AIAA Infotech@Aerospace 2013 Conference, Boston

[20] Sumant Tambe, Akshay Dabholkar, and Aniruddha Gokhale, "Fault-tolerance for Component-based Systems - An Automated Middleware Specialization Approach" In the proceedings of International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC 2009)

[21] Akshay Dabholkar, Abhishek Dubey, Aniruddha Gokhale, Gabor Karsai, Nagbhushan Mahadevan, "Reliable Distributed Real-Time and Embedded Systems through Safe Middleware Adaptation", In the 31st Symposium on Reliable Distributed Systems (SRDS), 2012, 362-371