ONLINE DISTRIBUTED VEHICLE AND MACHINERY HEALTH MANAGEMENT

Anthony Dietz, Eric Friets, William Finger, Jerry Bieszczad, Matt Miller
Creare Incorporated
Hanover, NH

Lawrence Freudinger
NASA Dryden Flight Research Center
Edwards, CA

ABSTRACT

Modern aircraft and space vehicles routinely sense and record vast quantities of information relevant to assessing the vehicles' health. However, limitations imposed by the bandwidth of telemetry and network connections prevent real-time transmission of the complete data set to central stations for analysis. An online health-management system suitable for bandwidth-limited network environments that enables interrogation of the full data set by ground-based operators is described. The system uses distributed objects organized in a data processing hierarchy linked by a buffered data-management subsystem. Reduced health information is routinely transmitted, but dynamic reports may be requested on demand from any object.

KEYWORDS

Vehicle Health Management, Condition-Based Maintenance, Networks, online Monitoring, XML

INTRODUCTION

Vehicle or machinery health management involves the use of sensed information from multiple sources to support decisions regarding safe or efficient system operation. Today's online networked infrastructure has the potential to revolutionize the field of health management, increasing system safety while reducing maintenance costs. However, the widespread application of online health management (OHM) systems is limited by the lack of tools for building systems capable of processing the vast quantities of data that are available from modern vehicles. Lack of high-bandwidth network connectivity to moving vehicles presents another challenge. In this paper, a network-centric approach to OHM system design and implementation that overcomes these challenges is described. Aircraft health management in the commercial aircraft industry is used as an example, but the approach described is relevant to a wide range of vehicle and machinery health management applications.

Health management applications can be divided into real-time and long-term time frames. Each serves different but complementary goals. Long-term trending information serves a goal of condition-based maintenance (CBM), where sensor information is used to observe slowly changing system properties from which the remaining useful life of components may be tracked and appropriate maintenance scheduled in a manner that minimizes aircraft or system downtime. For these purposes, sensor data are typically downloaded in intermittent blocks, such as at the end of a flight and processed some time later at a central location. The considerable savings that may be realized through such systems have prompted some industry action. Boeing is developing a prognostic and diagnostic Airplane Health Management (AHM) service [1]. Delta Airlines has just invested in a SmartSignal predictive maintenance system for engine health monitoring [2]. The length of modern flights and the rapid nature of current turnarounds make in-flight telemetry attractive even for long-term applications. However, it is in the time-constrained "real-time" role that in-flight telemetry and processing become critical.

Performance optimization and catastrophe avoidance both depend on the development of situational awareness within an adequate window of opportunity. On a short time scale, from seconds to hours, a health-monitoring system fulfills a safety role by detecting and reporting system anomalies and failures. An effective system goes beyond cockpit caution lights and processes and delivers situational awareness information to on- and off-aircraft personnel. These personnel are then in a position to make timely decisions and take appropriate action. The value of this real-time role is dependent on the quantity and quality of information that can be transmitted from the aircraft.

Current air-to-ground communications in commercial aviation are effected through the Aircraft Communications Addressing and Reporting System (ACARS). This is a VHF service provided by ARINC that transmits character-based messages at a low data rate. A digital link (VDLM2) is also available, which supports digital data rates of 31.5 kbps [3]. Boeing's Connexion system, recently introduced on Lufthansa aircraft, is a Ku-band satellite relay delivering network data rates exceeding 1 Mbps [4]. However, this system will also be utilized for cockpit data and passenger emails so only a reduced subset of the bandwidth will be available for condition monitoring. As with developmental test applications, the wireless data link remains constrained relative to potential uses. Value-added processing must be enabled on the aircraft to improve the quality of information exchanged with the vehicle.

The key goal of an OHM system is to get the right information to the right people at the right time. As the nature and context of each failure is different, the data set required to correctly diagnose a failure will vary on a case-by-case basis. This implies a need to enable more flexible and more comprehensive diagnostics and prognostics than are currently employed. In an ideal implementation, a remote troubleshooter alerted by the health-management system would be able to access the complete onboard data set in order to rapidly identify the problem. This is the goal of the approach we have taken to online health management.

REQUIREMENTS FOR AN EFFECTIVE OHM SYSTEM

An effective OHM system should meet certain key requirements:

- Combine information from a large number of sources over a distributed network.
- Analyze both historical and current data in real time.
- Implement system-specific decision-making models for prognosis and diagnosis.
- Report alerts to remote users.
- Allow scaleable investigation of the analysis and underlying data by remote users.

Prognostic and diagnostic algorithms will come increasingly from component manufacturers and from different research groups using various development environments. Sensor information and health-checking algorithms, which verify sensor performance, will come from the sensor manufacturers. Maintenance history and life information will come from the system manufacturer and the system operator. Furthermore, the capabilities of the health-management system itself will evolve over the life of the vehicle. Such a complex system-of-systems integration challenge can be met with an open, extensible, and collaborative OHM environment. Open standards are required to address the common needs of multiple organizations. An extensible approach is needed to allow operators to modify the OHM system as trends change, new information becomes available, and systems are upgraded. A collaborative environment is required as the resolution of specific failures may require the use of various experts with their own preferred application software. Finally, the next generation of the Internet will go beyond simple formatted presentation of data to incorporate *semantics*. In this semantic web, the *meaning* of information is conveyed through metadata (data about data), allowing software to perform increasingly adaptive processing. The expectation for OHM systems is to facilitate ever-increasing situational awareness in support of decision making.

AVAILABLE TECHNOLOGIES FOR AN OHM SYSTEM

Significant interest in CBM and OHM has resulted in the creation of a number of industry groups and emerging standards. One working group has developed the Open System Architecture for Condition-Based Maintenance (OSA-CBM), a seven-layer model CBM architecture encompassing data acquisition through to decision support and presentation layers, along with defined interfaces for each layer [5]. This work evolves from previously proposed standards including the MIMOSA CRIS and IEEE 1451. The aim is to move away from proprietary systems toward interchangeable modules with the associated benefits of reduced development time and more adaptable solutions.

online systems rely on middleware technology to provide communication services among objects distributed over a network. Recent middleware technologies include CORBA, DCOM, and Creare's Ring Buffered Network Bus (RBNB®). Of these, the latter technology was developed around the requirement for buffering data at various points in the network, a critical requirement for real-time telemetry and an important attribute for an OHM system [6]. Another applicable technology is XML, an extensible markup language that can be used to describe the data in a system in a form that can be interpreted by a range of programs and intelligent agents. Finally, a system with distributed processing capabilities requires a method to configure and initiate remote algorithms. RMI, SOAP, and XML-RPC are a few technologies that offer this capability. With such systems operating over networks, security is also a key concern to be addressed.

IMPLEMENTATION OF AN OHM APPROACH

System Overview

Creare has developed an OHM implementation approach that meets the requirements outlined above. The OHM software enables configuration of health-management systems using object-oriented sensor and algorithm components via an intuitive graphical user interface. Objects are connected in a multi-layered hierarchical data processing structure, which may be distributed across a network. The data and data structure are described by a relational metadata schema. As a result, health reports may be dynamically generated from data at the system, subsystem, or sensor level. Furthermore, reports and data may be viewed online with a standard Web browser. The software is built upon RBNB middleware software, which provides local data buffering and enables online processing, communication, and administration of current and historical data in real time. An example implementation is depicted in Figure 1. The ability to configure and initiate remote algorithms that can consolidate raw sensor data into relevant health information at each level of the hierarchy significantly reduces network bandwidth requirements. Simultaneously, the data accessibility is increased through distributed data buffering and Web-based report generation.

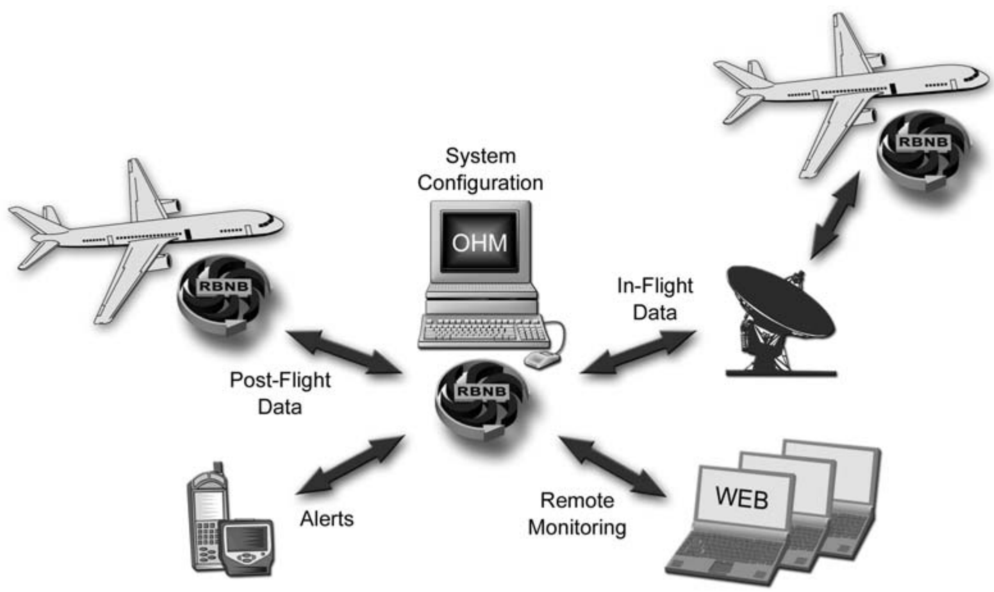

Figure 1. Buffered Data and Dynamic Reporting Ensure Telemetry and Network Loads Are Minimized During Prognosis and Diagnosis of Vehicle Health.

System Configuration

The OHM implementation involves two main software tools: the Object Configuration Tool (OCT) and the Online Reporting Tool (ORT). An example configuration is shown in Figure 2. The OCT provides a graphical workspace in which sensor and algorithm objects can be connected to form a distributed data-processing structure. Sensor objects represent sensor data channels on RBNB servers. Algorithms are programs configured to process data from sensors or from virtual sensors (other algorithms), and to write the results back as other RBNB channels. XML-RPC messages sent over the RBNB network are used to configure and start the algorithms via the Algorithm Factory, an RBNB plugin utility. When the system configuration is complete, the data-processing structure is executed causing data to start flowing through the structure. The OCT then generates an XML metadata file completely describing the hierarchical data-processing structure.

The second tool, the ORT, provides a reporting capability that can be accessed with a standard Web browser. It is located in the DMZ outside the secure data acquisition network and interfaces through the installed security system. The tool uses the XML metadata generated by the OCT to provide a navigable graphical representation of the data-processing structure in HTML Web pages. Users may drill down to any level of the structure and view system-, subsystem-, or even sensor-level pages. These pages are compiled by the ORT using data from onboard RBNB servers and graphics generated by onboard utilities. With this as-required approach, remote users can access the entire data set and associated analyses without unnecessary network traffic. This bandwidth-conserving design feature aids scalability. Each of the components is described in more detail in the following sections.

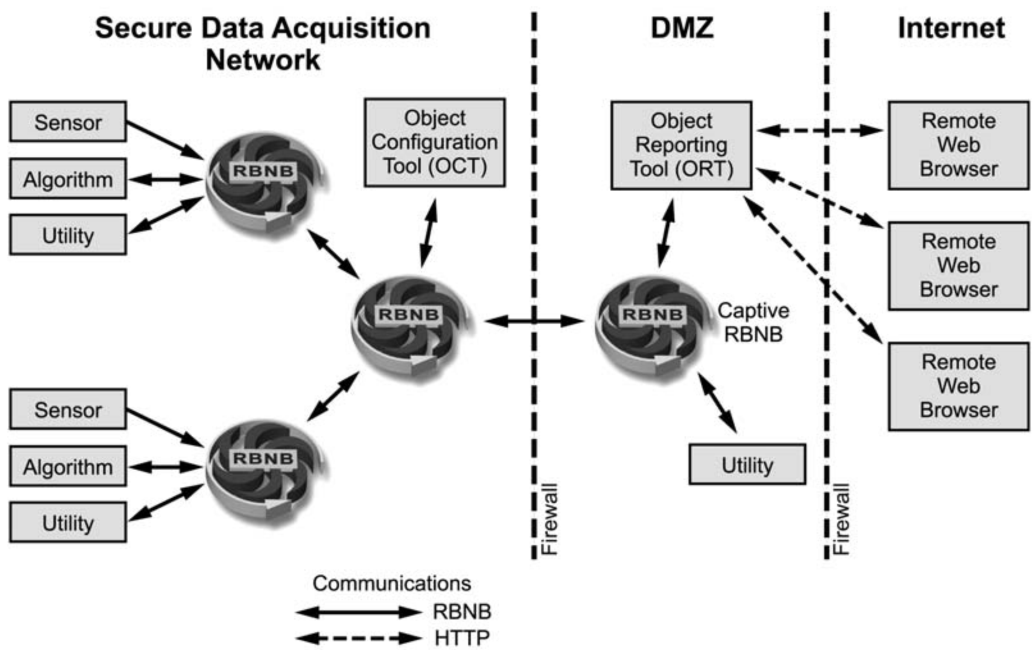

Figure 2. Example OHM System Configuration. RBNB Servers form the backbone of a data-processing structure containing sensor and algorithm objects. A system designer configures the system with the OCT. Remote users access the data via the ORT.

RBNB Network

The RBNB network is the backbone to which the various OHM modules interconnect and communicate. Sources such as sensors "fire and forget" data to the RBNB network; i.e., they do not directly send data to particular clients (sinks). Data sinks access content from the RBNB network buffers, not directly from the sources. Historical information is buffered in the network itself, such that origin sources have no responsibility to ever retransmit data, and individual sinks need not continuously monitor and locally store data. Plugins are "smart" virtual channel data sources, in that they process other data channels on demand. Critical storage and bandwidth resources are managed with great flexibility and efficiency via the placement and topology of the interconnected RBNB servers.

Object Configuration Tool

The OCT facilitates the configuration of an OHM system by providing a graphical workspace, in which objects can be dragged, dropped, configured, and connected. Figure 3 shows the OCT user interface. On initiation, the tool executes a discovery of the RBNB network and identifies the hierarchical structure of the nodes and the available sensor and algorithm resources. These are displayed in a resource library (left frame) where objects can be organized by type, IP address, RBNB router, etc. Such different views help the system designer gain a picture of where resources are located in the network, so that potential networking and data-processing bottlenecks will be avoided. Resources can be dragged onto the workspace and connected to form a logical data flow diagram. This diagram gives the designer an intuitive picture of the flow of data through the objects in the system. Objects include sensors, algorithms, and alerts. Sensor Objects represent sources of measurement data from physical systems. Algorithms represent data-processing elements that take one or more input data streams (from sensors and/or algorithms) and generate one or more output data streams. Alerts are a form of algorithm representing data consumers, to which the high-level outputs of algorithms are pushed immediately, rather than queried using the ORT. For example, this may represent personnel who are paged or emailed when critical information is generated, or other personnel who receive routine reports. Objects can be configured in a multi-layered hierarchical structure by selecting a set of objects and grouping them and their interconnections as “subsystems.” The subsystems are then accessed in a separate Subsystem Decomposition Window (lower frame). The hierarchy of objects is depicted by a directory tree in the right upper frame. Finally, the XML metadata associated with a selected object is shown in the lower right frame. Although the metadata is shown in the raw XML format, it may be formatted by including an XSD style sheet. These metadata assist the designer by describing the object and the data it represents. Specific objects may be located by searching for their associated metadata.

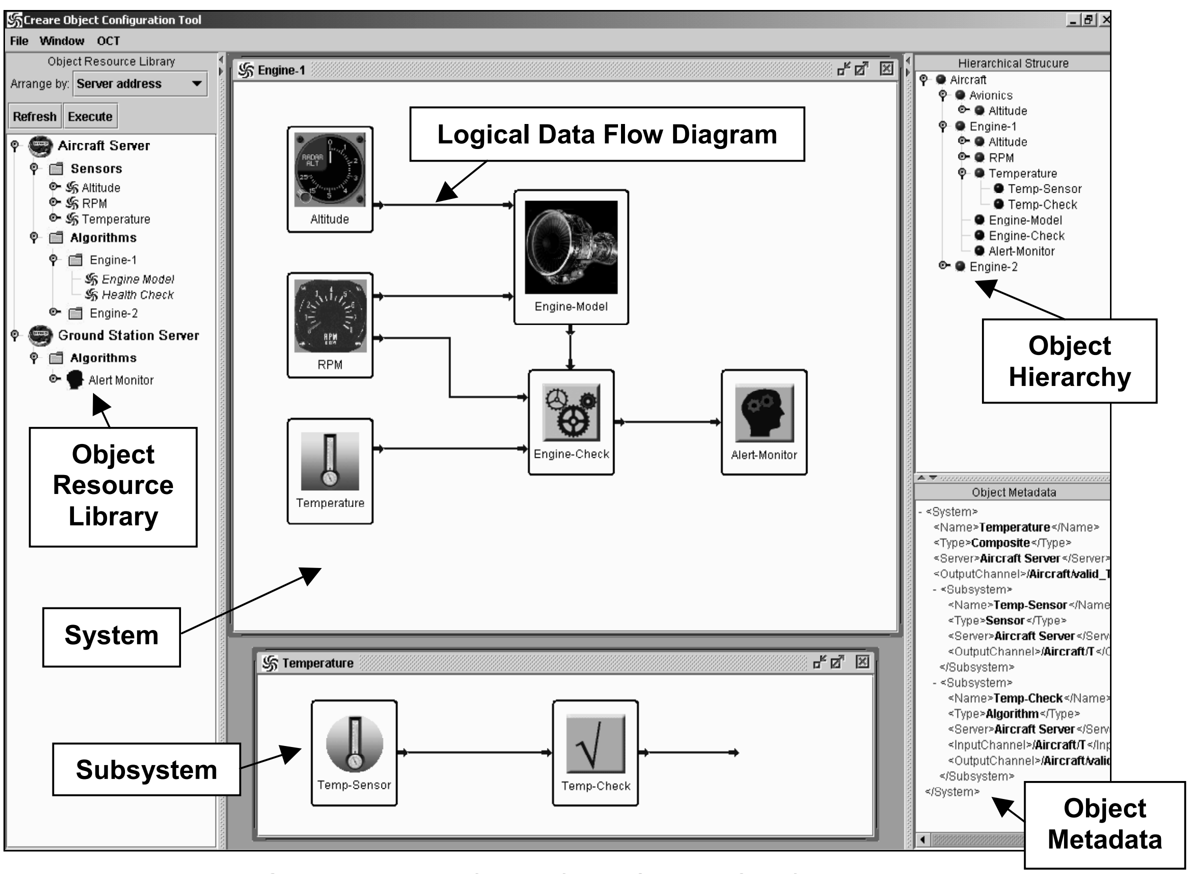

Figure 3. Graphical Interface of the Object Configuration Tool

Remote Algorithm Configuration and Initiation

When configuring a system with the OCT, algorithm objects are selected from an Algorithm Factory, which is loaded onto remote machines before configuration begins. The factory is an RBNB plugin, which acts on instructions containing XML-RPC commands to invoke, configure, and then start algorithms. Early prototypes of the OHM used Remote Method Invocation (RMI) to implement this function, but the XML-RPC approach provides a more efficient solution as all communication occurs over the RBNB network. Once the system has been completely defined, the designer initiates its execution from the OCT. At this point, the remote algorithms are started and data starts flowing through the system. The OCT also writes an XML file completely describing the data processing structure.

Online Reporting Tool

The ORT tracks the failures and status messages generated by the system and reports a ranked list of messages on a separate Web page. This is equivalent to the standard reporting capabilities of current OHM systems. However, the ORT also delivers a powerful level of flexibility to the OHM system. With typical data-pipeline type systems, the user is limited to the data set supplied at the end of the pipeline. However, the ORT provides remote users with the ability to browse through the data structure and drill down to any level of information, from high-level status inferences to low-level raw sensor data. It does this by drawing on the metadata created by the OCT, which fully describes the data-processing structure, including the logical diagram graphics, the network nodes, and the channel names of each object in the structure. The ORT uses this information to dynamically create Web pages as users click down through the structure. As shown in Figure 4, each page shows the logical data flow for the particular system or subsystem being investigated. The full hierarchy is shown in the frame on the left. When the user reaches sensor or algorithm levels, plots of the output data are presented. Complex reports of different levels of a system are compiled using the shopping-cart type form at the base of the screen. These forms are saved with other individual user information in the RBNB. Data sets can be downloaded in a similar manner.

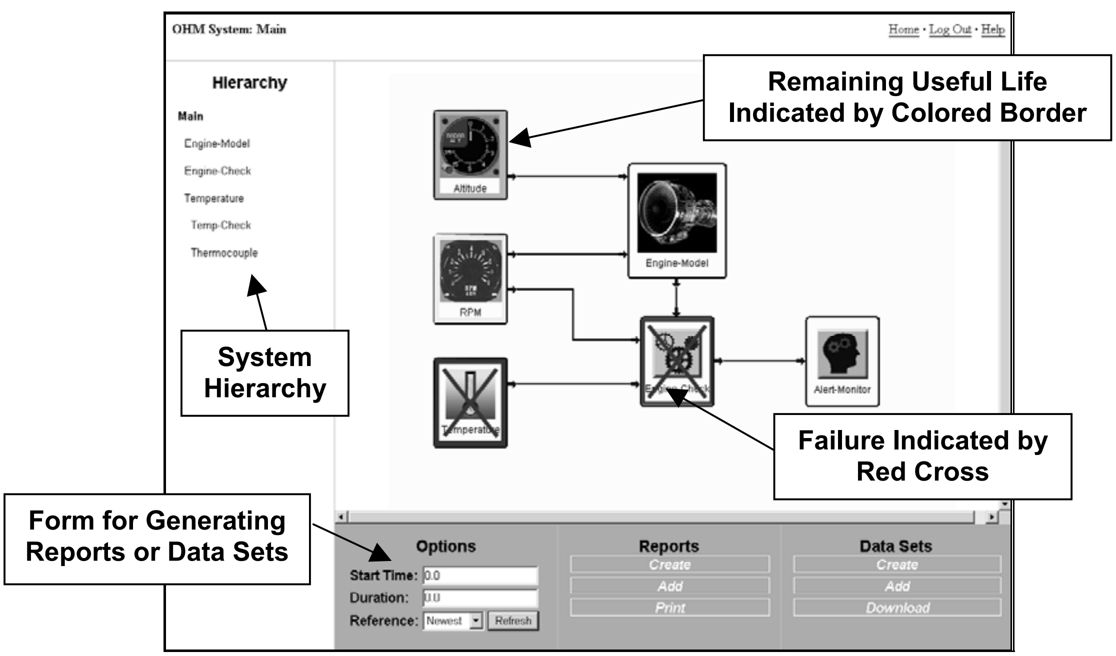

Figure 4. Web Page Published by the Online Reporting Tool

Health and Status Tracking

Each health-management system will contain specific health-checking algorithms to report on the health of subsystems and components. The output of these algorithms includes a pass/fail channel and a remaining useful life (RUL) channel. A list of these algorithms is included in the metadata produced by the OCT when the system is configured. The ORT monitors these channels and compiles a ranked list of system status and failure events. These are reported on a top-level web page. Failed objects and the higher-level objects containing failed objects are also indicated in system and subsystem pages by a red cross on the object in the logical diagram and by red text in the hierarchical tree (Figure 4). The RUL of objects is indicated by colored borders about the object, which progress from green to red as the object's life is used up. The life of a higher-level object is equal to that of its subsystem with the shortest life. RUL is useful for planning maintenance, whereas failures are more emergent actions that must be addressed for safety reasons. For example, a tire may be bald (RUL = 0%) or a sensor may be two months away from its annual calibration (RUL = 16%), while still not having failed. However, a sensor may fail while still having significant RUL.

System Security

Security is an important component of any trustworthy OHM system. The current implementation described here employs Web-standard Secure Socket Layer (SSL) technology to authenticate users and encrypt data as it traverses the network between the ORT and the remote browser and to prohibit direct access to the underlying RBNB network. SSL support is built into all client browsers, minimizing the burden on users while maintaining strong security. The RBNB network and the OCT are designed to reside on a private network protected by a firewall. In situations where this private network needs to be distributed over two or more sites, a Virtual Private Network (VPN) may be used. To enable secure, outside access, the ORT and companion RBNB will be installed outside the firewall, with an RBNB routing connection across the firewall. The companion RBNB will be configured to allow connections only from the ORT and machines inside the firewall. Hence, the only access external users have to OHM data will be via the ORT. This configuration was shown schematically in Figure 1. This approach uses standard, commercially available security products (e.g., firewall, VPN) to protect the OHM system and ensures that an OHM installation can be configured to meet site-specific security requirements.

THE KEY ROLE OF METADATA

The Semantic Web

Today's Internet is limited by the capabilities of its markup language, HTML. This language only describes how to format data for presentation; it does not describe the data itself. The next generation Web will use semantic metadata (data about data) to describe its contents. When data is accompanied by metadata, intelligent processors or agents can interpret the data and make decisions based on the content. This revolutionary development will be enabled by systems that generate metadata along with data content, and by processes that can interpret and act on this metadata. The use of standard metadata formats will allow for scaleable and collaborative applications with independent designs that are able to share data. The OHM Software implementation described here is an early example of such a system.

XML

The OHM design relies on XML, which is a markup language that encompasses each piece of data content in tags that describe the actual content. XML metadata is used to describe the sensor and algorithm objects and also to describe the hierarchical data-processing structure itself. It is this metadata description that enables a designer to locate, connect, and configure sensors and algorithms and a user at a standard Web browser to navigate through the data flow structure. As a result, specific metadata formatting is required.

Schema

Metadata formats or structures are known as schemas. A schema defines the type of data that may be associated with a particular tag and the relationships between nested tags. The schema may be used to check XML documents for conformity prior to processing. As mentioned earlier, there have been a number of attempts to define schemas for sensors and data acquisition applications. Two relevant attempts are SensorML developed by OGC for earth-sensing applications [7] and the OSA-CBM schema developed for health-monitoring applications [5]. Our challenge in adopting these schemas is to identify the elements with the specific structure we require.

OHM Schema

We have defined an OHM-specific schema for the data-processing structure as it is specific to our application. However, the open nature of the schema and associated metadata will allow other programs to interrogate the data-processing structure. We have also defined a schema for the sensor and algorithm metadata. This is a minimal schema containing only those elements required to manage the data. Elements in a broader schema may be used if one emerges as the industry standard, or translation algorithms may be employed to identify the OHM-specific elements in other assorted schemas. An important difference between our approach to metadata and that presented in some of the current schemas is that we store the data and the metadata in separate channels (or files). The two are linked by naming conventions or by reference in the metadata. Other schemas insert the data into the XML document. Our approach is necessitated by the need for real-time buffered data. Such time series data are continually being generated and old data are archived or lost once the buffer is filled. On the other hand, metadata are typically static or slowly changing relative to data. Separating the two forms of data increases processing speed and reduces memory requirements.

EXAMPLE IMPLEMENTATION

An example implementation of an OHM system could be the aircraft health-management scenario depicted earlier in Figure 1. Each aircraft is configured with an RBNB data server containing data from the aircraft bus and from specific health-monitoring sensors. Each data channel has an associated metadata channel. The metadata are generated from the transducer electronic data sheet (TEDS) as defined by the IEEE 1451 standard or from other standard sources. The OHM system designer would use the OCT to perform an initial discovery to locate the server and its available data channels and algorithms. The designer would then configure the system, using onboard algorithms to process the data into health information, and then ground-based algorithms for long-term trend information and fleet comparisons. He or she would set up alerting algorithms for time-critical warnings and routine algorithms for maintenance information. Finally, the designer would execute the system, at which point the data would start flowing through the processing structure. The system design could then be loaded onto other planes in the fleet. An engineer, alerted by a vibration health-checking algorithm, would use a Web browser to access the system health Web page published by the ORT. He or she could then inspect a report listing the system failures and drill down to the level of the component experiencing abnormal vibration. The failure might have been generated by a specific FFT algorithm processing data from a particular accelerometer. The engineer could inspect the FFT trace and also the raw signal trace from the accelerometer. The intermittent nature of noise on the accelerometer, indicative of an electrical fault, might be enough to convince him or her that an emergency landing is not required. This scenario illustrates the advantages of the flexible nature of this OHM implementation. It would be impossible to transmit all of the information that was required to make this diagnosis to the ground in real time with current or even next-generation aircraft data links.

CONCLUSION

An approach to online health monitoring of vehicles or machinery that overcomes telemetry and network bandwidth limitations was described. This approach provides remote users with access to the full sensor data set in near real time. Distributed objects are organized in a data-processing hierarchy and linked by a buffered data management system. Reduced health information is routinely transmitted, but dynamic reports may be requested on demand from any object. The approach remains open, extensible, and collaborative through the use of RBNB middleware and XML metadata. The RBNB middleware provides a data buffering capability and dynamic file servers that may be accessed by any programs. The XML metadata, used to describe the data-processing structure and the raw and processed data channels, can be interpreted by other programs. Key requirements of access to real-time and historical data and flexible reporting are also met through the use of these two technologies.

ACKNOWLEDGMENTS

This work was supported by SBIR funding from NASA Dryden under contract NAS4-02039.

REFERENCES

1. Mecham, M., "When Will It Break? Boeing Seeks Answer," Aviation Week and Space Technology, Vol. 158, No. 1, Jan. 6, 2003, pp. 39–40.
2. Ott, J., "Engine Faults Flagged In Predictive System," Aviation Week and Space Technology, Vol. 158, No. 7, Feb. 17, 2003, pp. 54–55.
3. Mecham, M. "Boeing Makes Its Internet Connexion," Aviation Week and Space Technology, Vol. 157, No. 26, Dec. 23, 2002, pp. 42–43.
4. http://www.arinc.com/product/voice_data_comm/vdlm2.html (May 5, 2003).
5. Lebold, M., "An XML-Based Implementation of the OSA-CBM Standard Using SOAP Over HTTP," Proceedings from the 57th Meeting of the Society for Machinery Failure Prevention Technology, Virginia Beach, VA, Apr. 14–18, 2003.
6. Miller, M. and Freudinger, L. "A Web Compatible File Server for Measurement and Telemetry Networks", Proc. 2003 International Telemetry Conference, Las Vegas, NV (to be presented).
7. Botts, M., and McKee, L., "A Sensor Model Language: Moving Sensor Data Onto the Internet," Sensors, Vol. 20, No. 4, Apr. 2003, pp. 30–33.