Software Architecture for Condition Monitoring of Mobile Underground Mining Machinery

A framework Extensible to Intelligent Signal Processing and Analysis

Jordan McBain, Markus Timusk
Bharti School of Engineering
Laurentian University
Sudbury, Ontario Canada
mcbainjj@gmail.com, mtimusk@laurentian.ca

Abstract—In the mining sector, there are growing calls for the ability to monitor the health of operational assets like structures, mobile underground machinery, and complex stationary equipment. This work focuses on the development of a software architecture to monitor the entire range of industrial and mining equipment – all the while acknowledging the more generic problem of intelligent signal processing having applicability to a much broader class of problem. The implementation of condition monitoring for mobile underground mining equipment relies on previous work by the authors in advancing condition-monitoring techniques for variable speed and load machinery. The design will permit flexible run-time system configuration with a variety of choices in signal processing and intelligent analysis techniques; it has been demonstrated to work well with an implementation in MATLAB object-oriented programming (OOP) from data collected on a real gearbox subject to varying loads and speeds. This success justifies the advancement of a full-fledged prototype in LabVIEW OOP. The aim is the development of the data analysis layer; the result should integrate seamlessly with such developing industrial standards as the International Rock Excavation Data Exchange Standard (IREDES).

Keywords—Condition Monitoring; Variable Speed and Load Machinery; Service-Oriented Architectures; IREDES

I. INTRODUCTION

Mobile mining equipment is critical to the operational success of most mines. Despite its importance, there are few techniques for in-situ online monitoring of the health of its components and ultimately diagnostics and prognostics of potential points of failure. This class of machinery is subject to time-varying conditions primarily through variations in speed and load; condition-monitoring of this class of machinery is a challenging problem requiring the application of a diverse range of theory – potentially including adaptive signal processing, system identification, artificial intelligence, etc. Automated solutions for this class are a growing point of concern in the literature.

Regardless of the difficulty in automating the online detection of faults in time-varying equipment over that of equipment subject to stationary conditions, there are strong commonalities in the steps employed to achieve this functionality. Moreover, the procedures in condition monitoring of machinery are similar to those in structural health monitoring, process monitoring, measure-while drilling (a process to detect the type of material a drill is penetrating based on signals from the drill), seismicity analysis, etc. Clearly, the more abstract problem of "intelligent signal processing" frames the present condition-monitoring concern – ranging from the monitoring of biomedical applications, maritime propulsion and auxiliary systems, aircraft systems, to sensor-failure diagnostics and beyond (see Figure 1).

The focus of this work is a low-level software architecture capable of monitoring mobile mining equipment in order to detect faults and potentially diagnose and prognosticate its remaining usefulness. Design for change is the primary objective in developing this architecture – consequently the entire range of applications described above is considered but the final product focuses on condition-based monitoring (CBM). While not the primary task, the authors further investigate the importance of enterprise architecture standards like the International Rock Excavation Data Exchange Standard (IREDES) for this type of architecture.

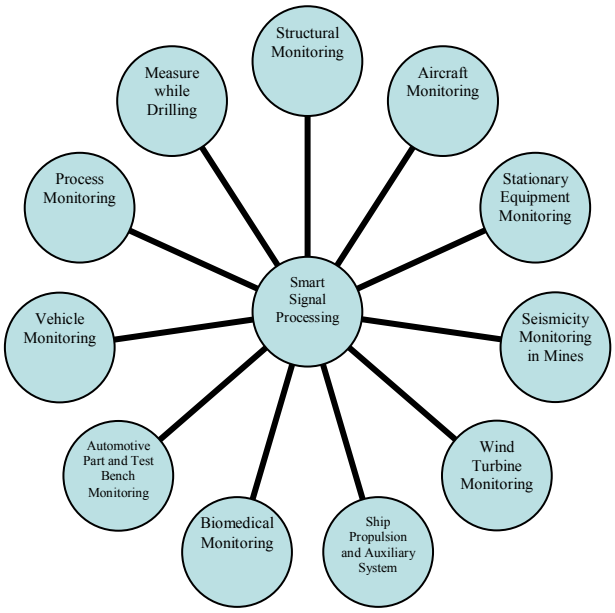

Figure 1 Breadth of System Applicability

978-1-4673-0358-3/12/$31.00 ©2012 IEEE

II. BACKGROUND

A. *Condition Monitoring of Mobile Mining Equipment*

Monitoring machinery for early failure detection is an established practice in industry; where the benefit associated with the ability to detect incipient faults in critical operating machinery justifies the cost of purchasing monitoring equipment, substantial returns on investment can be generated with this predictive maintenance strategy. For monitoring stationary equipment, there are wide arrays of methodologies that are entrenched in practice and in the literature. While vibration analysis is more predominant, other methods such as oil analysis, thermography, and other non-destructive solutions are available.

A developing area of research involves fault detection in time-varying systems. In the mining sector this includes mobile underground mining equipment, hoists, electro-mechanical shovels, etc. These systems are generally subject to varying load and speed and consequently, unmodified traditional techniques tend to be of limited value. The spectra of vibrations in this type of equipment are non-linear in nature and the signature of faults in those spectra will not be easily discerned. Their characteristics are masked not only by resonances derived from speed changes but also frequency- and amplitude-modulation effects from changing mesh stiffness and varying loads. Machinery subject to these time-varying conditions is of primary interest to the researchers.

Automating the detection of faults with stationary equipment or this more complicated class is an imperative problem. Eliminating the need for human analysis by mimicking human intelligence with artificial intelligence provides a feasible means of achieving this aim. For example, one might seek to automate the traditional technician's fault-finding technique that relies on the observation of a screw driver set atop a vibrating machine with these computational algorithms.

Typically, data representing a machine's state is available in abundance only for the healthy state. It is either too difficult, operationally and/or economically infeasible to develop data representative of each possible (or any individual) failure mode. This limits the use of many artificial-intelligence techniques that might generally rely on a complete description of the problem space. Novelty detection, however, provides an ideal solution to the problem. This class of algorithms provides a boundary that delineates a machine's state from healthy and faulted. The boundary is developed by minimizing a curve around a set of healthy feature vectors gathered during training that scatter in an n-dimensional Cartesian plane. There are a large variety of algorithms available for this problem (see [1,2])

Figure 2 demonstrates novelty detection achieved with the Support Vector Data Descriptor (SVDD). The SVDD fits a minimal-radius sphere around data sampled from a "normal" class after it has been augmented into a high-dimensional space. Once the computed sphere is re-mapped into normal n-dimensional Cartesian space it creates a highly non-linear and non-convex boundary employed to discern between normal and abnormal test data (consider, for example, Figure 2 (A)). A novelty score is generated, as shown in Figure 2 (B), by measuring distance from the novelty boundary; a sampled data set's distance from and distribution with respect to the novelty boundary is an important indicator of the quality of the novelty detector's classification results (superior results are achieved when the data scatter further from the boundary or, in the advent that they scatter closely, if they have small variance).

One of the primary thrusts in accounting for the variable-state nature of this problem focuses on adapting the novelty boundary (see [3-7]). The selection of the feature vector used to describe the problem is often more critical than the choice of novelty detection approach. In monitoring variable-state machinery, this choice is particularly potent; choosing feature vectors that are insensitive to the variable-state nature of the domain is another predominant approach in the literature (see for instance [8-10]).

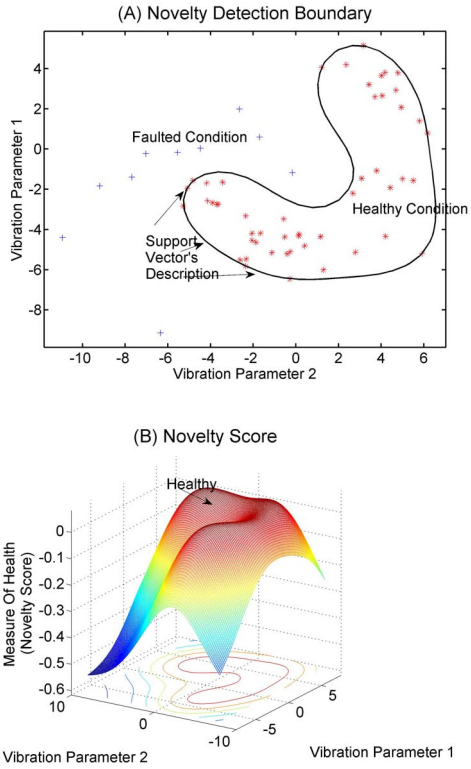

Figure 2 Novelty detection (a) boundary used to classify machine state and (b) novelty score commensurate with positive and negative distances from the boundary (adapted from [4] **and generated with** [11]**)**

In an upcoming work, the authors demonstrate the simple step of including one modal parameter in the feature vector when analyzing faults from a gearbox subject to varying loads and speeds (see Figure 3 and Figure 4). This test bench features a 50-hp and 25-hp motor driven by variable frequency drives to independently control the load and speed profiles experienced by a single-stage gearbox (further

described in [12]). A variety of vibration-centric feature vectors such as auto-regressive (AR) models, standard acoustic emission features, and statistical parameters were combined with speed to discern their ability to delineate faulted states. The classification results using AR models, generated from vibration signals gathered from the gearbox's bearings, without any consideration of speed or load is poor as shown in Figure 5. After adding speed to the feature vector (composed of AR models), the results improve but are not ideal (see Figure 6); when the gearbox is subject only to varying speeds, the classification results are excellent – the additional problem of varying load requires a more advanced solution.

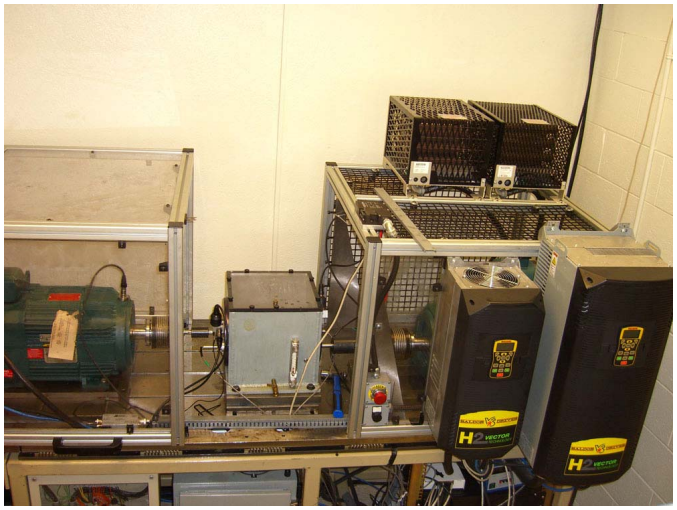

Figure 3 Test bench (25 hp motor (left), gear box, 50 hp motor partially obscured with VFD's for motors, control and data acqusition below VFD's)

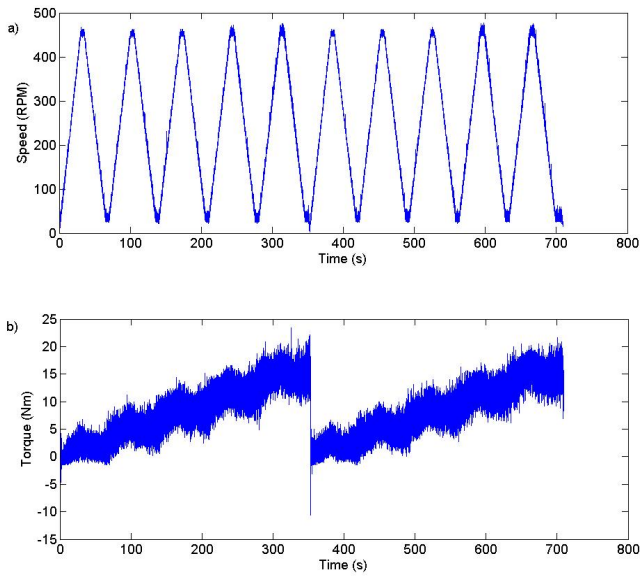

Figure 4 Typical (a) speed and (b) load profile employed during training

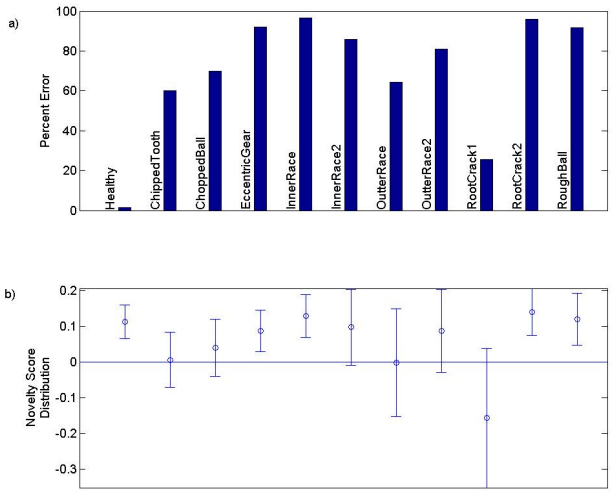

Figure 5 (a) Classification results using an order-20 autoregressive model with standard novelty detection (no consideration of speed or load) with (b) respective novelty score variance

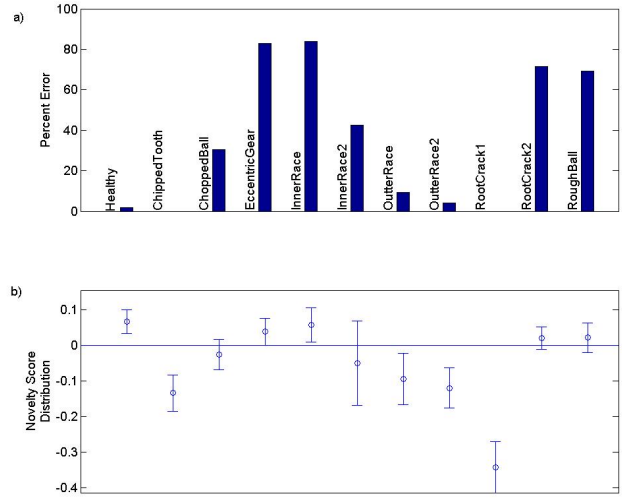

Figure 6 (a) Classification results using multi-modal novelty detection with only speed considered with (b) respective novelty score variance

Acknowledging that the vibration from this class of machinery is non-linear in nature, the authors postulated that its behavior might still be characterized with linear system identification methods when coupled with novelty detection. By employing a gearbox's input speeds and loads as a system's input and the same gearbox's bearing vibration as a system's ouput, a MIMO model was built with ARMAX-based system identification. When the transfer functions developed with this model are used in the feature vector,

classification of bearing and gear faults in time-varying machinery is demonstrated to be excellent (see Figure 7). This approach is shown to generalize well across ranges of speed and load not experienced during training. The results have been submitted and are under review; a similar approach can be found in [10].

Regrettably both these methods require the measurement of the time-varying parameters. Tachometers and load cells are both expensive and can be difficult to implement on established machinery in industry. A simplified and more powerful technique lies in correlating the signal from accelerometers (or other sensors) that are located on different components of a machine. For instance, when correlating the vibration signal from accelerometers on the input and output bearings of a gearbox, we expect that parameters of the resultant signal would be unaffected for changes in load and speed; when faults begin to affect the signals, they should do so in an uncorrelated fashion and the behavior should deviate accordingly. By fixing an AR40 model to the cross-correlation signal from the vibration signals of the inputs and outputs of gearboxes, faults are just as easily identified when the gearbox is subject to time-varying conditions as demonstrated in Figure 9. The technique keeps the earlier identified generalization properties but omits the difficulty and cost associated with load and speed signals as well as the computational demands of advanced system-identification techniques. These results will be published in an upcoming journal.

B. International Rock Excavation Data Exchange Standard

The International Rock Excavation Data Exchange Standard (IREDES) Online is an XML-based communication schema designed to make data exchanges generated by common classes of mining machinery the same, irrespective of manufacturer. It enables the transmission of real-time data so that mine operators or managers can affect control or appropriate operational changes. IREDES is designed to be an industry standard that is universal, interoperable, and extensible [13].

It has two primary communication models: one where the client connects directly to the machine and one where a server is employed as a middle man. In situations where an embedded machine's computational capacity (e.g. data storage, processor speed, etc.) is limited, the latter is preferred. Present machine status may be requested in either case or where a reservoir-like capability is provided, either on the machine or through a server, historical data may be provided. The online IREDES standard facilitates a read, subscribe and historical report functionality – respectively, they enable the user to get some aspect of the current status of the machine (e.g. speed), to subscribe to a machine's state changes (e.g. change in position), and to get an historical report of the machine's data for a given period (e.g. position) [13].

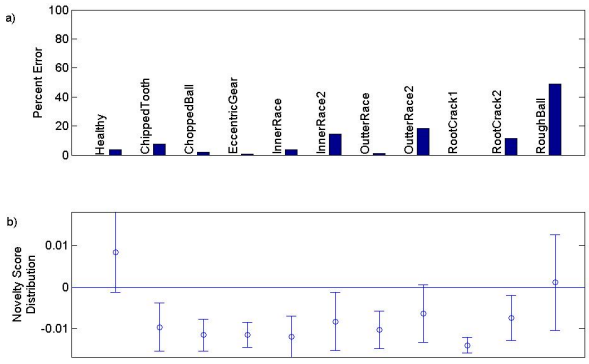

Figure 7 (a) Classification results with system identification (10 Zeroes and 20 Poles) with (b) respective novelty score variance

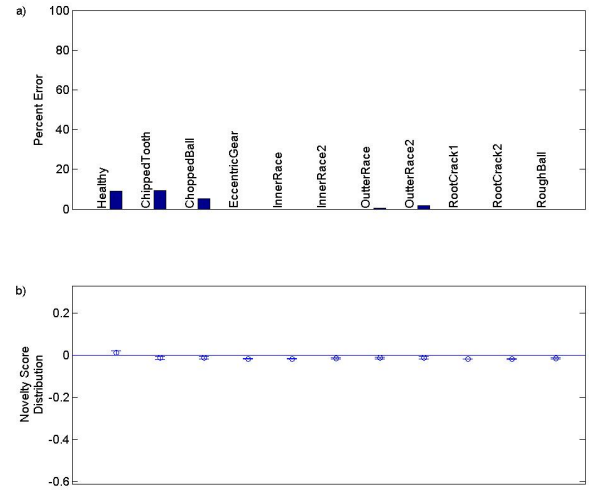

Figure 8 Classification results (a) with standard novelty detection based on an AR40 model of the cross-correlation of vibration signals from two disparate accelerometers with (b) novelty score variance of classification results

These functions can access generic properties of a machine. The properties of a machinery class may be modified or amplified to create more specific features all the while maintaining the common interface used across machine class instances [13]. At present, it enables the reporting of production performance and quality as well as the transmission of product plans. Future specifications will enable the reporting of machinery health status.

C. Literature Review: Software Architecture for Condition Monitoring

A number of researchers have defined architectures with varying degrees of abstraction. Modern systems focus

primarily on standards such as the Open Systems Architecture for Condition-Base Monitoring (OSA-CBM) and its progenitor ISO 13374. These standards and more concrete architectures in the literature are reviewed to provide a baseline for the system proposed herein.

The work in [14] identified deficiencies in the connection between data from predictive-maintenance strategies and upper enterprise data; it argued for improvements in information integration and exchange through the consolidation and condensation of data for upper-layer systems in enterprise resource planning (ERP). To this end, they highlight the abstract details of a concrete model of data between actual equipment, the predictive maintenance framework and the enterprise system. Similarly, Rao [15] divides the problem into four modules: data calibration, condition monitoring, fault diagnosis, and maintenance assistance.

In [16], a more concrete software architecture was specified that employs fault trees, fuzzy logic and novelty detection – the focus in this work is the actual implementation of a condition-monitoring system rather than its integration with ERP. Similarly, in [17-19] an architecture for data fusion of sensors for condition monitoring is exposed; similar to other works, it models generic data flow through preprocessing, data alignment, and post-processing strategies. Jackson [20] describes a distributed grid architecture for real-time pattern matching analysis with data mining for EHM. In [21], Zhang describes a pattern-oriented software design for analyzing asset health; the work is similar to this one but with a focus on implementing existing static design patterns rather than ones permitting dynamic system configuration.

ISO 13374 defines the generic data flow of a typical condition monitoring system. Data is transformed into information from data acquisition (transducers) to data manipulation (signal processing) to state detection (abnormality search) to health assessment (fault diagnosis) to prognostic assessment (remaining useful life) to advisory generation (alarms) (see Figure 9).

The OSA-CBM standard is one fulfillment of ISO 13374; it was developed by a consortium led by Boeing, the US Navy, Rockwell Automation, Caterpillar, etc. Like the ISO standard it aims to simplify the task of integrating software and hardware involved in condition-monitoring systems into a standard architecture. "In short, it describes a standardized information delivery system for condition based monitoring" [22]. The most recent incarnation is specified in the Unified Modeling Language (UML) enabling the separation of information exchanged from the individual vendor implementations of hardware and software technology. It can be implemented in any middleware (software enabling the integration of two disparate applications) programming language such as Web Services, CORBA, etc. OSA-CBM may be implemented with OSA-EAI which is geared more toward the distributed database-driven information management activity and it "feeds up" into the full enterprise application integration level [23].

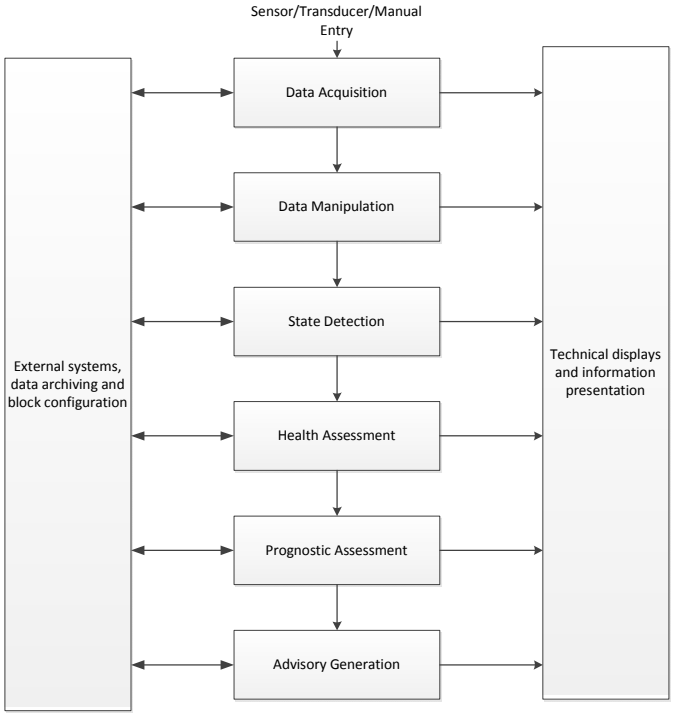

Figure 9 Information processing model for condition-based monitoring and diagnostics based on ISO 13374

Chidambaram [24] attempted to implement OSA-CBM for a system monitoring a brushless DC motor driving a hydraulic pump. It was further investigated with an in-vehicle health monitoring (IVHM) system by GE aviation in [25]. They championed the standard because of its ability to support a modular and scalable infrastructure at a time when it was complex to interface disparate commercial equipment health monitoring (EHM) systems; absent OSA-CBM, this constraint was further deemed as one of the predominant limiting factors preventing the widespread use of EHM systems. Notwithstanding the flexible standard that OSA-CBM affords, [25] identified experience at GE indicating that not only is it impractical to capture sensor data in bulk during flight with post-processing but that it is also equally impractical to process all data in real-time during flight. They further identified difficulties in monitoring different parts of a plane "where necessary for safety" due to concerns from weight (from excessive wiring), and perceived costs associated with multiple monitoring systems (when it can cost on average £ 90,000 to monitor a helicopters drive train alone). Another system for IVHM can be found in [26].

A similar effort can be found in [27] proposing an architecture where "agents" (intelligent technology for software that acts on behalf of a client) are used for EHM. The developed agents purportedly can imitate the logical thinking process of an expert assessment, including acquiring information, deducing, analyzing and making decisions. The authors specify a hierarchy; at the lowest level one class of agent gathers and processes in-situ data from sensors; subsequently a management agent correlates decisions from lower levels to generate a more realistic health assessment

that feeds a human-computer-interface (HCI) agent whose task is the presentation of results to the user. This abstract architecture was fitted to a monitoring system that monitors long-span bridges. Another analogous coarse-grained architecture is described in [28].

Efforts similar to OSA-CBM can be found in [29,30]; these works describe the importance of service-oriented architectures (SOA). SOAs are the software industry's response to network applications developed in silos; their aim is to permit reuse of network application functionality and internal information. Standards like OSA-CBM are the mortar that holds these network applications together.

III. Software Capabilities

An analysis of the proposed system's operational capabilities, implementation environment, range of expected uses, and potential interfaces is considered before reviewing the software architecture. An idealized solution is displayed in Figure 10. The focus of this architecture is on the low-level process and analysis of machinery signals; it must ultimately respect such established standards as OSA-CBM.

A. Vision

In this work an intelligent signal processing software architecture is viewed as one that takes real-world signals and then processes, segments, extracts relevant information and classifies it while enabling decision-support for the given problem domain (see Figure 11) [31]. The framework should support flexible and dynamic routing of signals through each stage of this process to enable future extension with new algorithms and user-directed data flow (possibly to external systems).

For health monitoring of machinery, this flexibility should ultimately enable the collection of signals from a variety of transducers (such as accelerometers, acoustic-emission sensors, tachometers, load cells, thermocouples, etc.) that are subsequently segmented into discrete and coherent analysis intervals used to extract feature vectors that are fed into pattern-recognition algorithms (or other fault-detection algorithms such as expert systems) and post-processing including diagnostics, prognostics, sensor-failure detection and reporting. The choice of the paths employed in solving this problem should not be hardcoded but left to a condition-monitoring expert at run-time for configuration for use in a wide array of monitoring problems. Once configured, the system should support reporting to alarm-management systems, interfacing with industrial databases (e.g. PI data historians, Wonderware, etc.), and remote user monitoring (e.g. from IPads, network terminals, etc.).

While intelligent signal processing is the aim in this work, it should be noted that much value could be added to in-vehicle health monitoring (IVHM) systems and those monitoring stationary underground mining equipment through the availability of simple reporting for other problem domains. The reporting of the position and operations of mobile-mining equipment could add value to mine-planning systems when fully integrated at the enterprise level. More specifically, the provision of environment monitoring data and underground activities could greatly augment automated ventilation on-demand systems. These contemplations are dog-eared for consideration in the interface specification so as to leave their later implementation a simple exercise – designing for change.

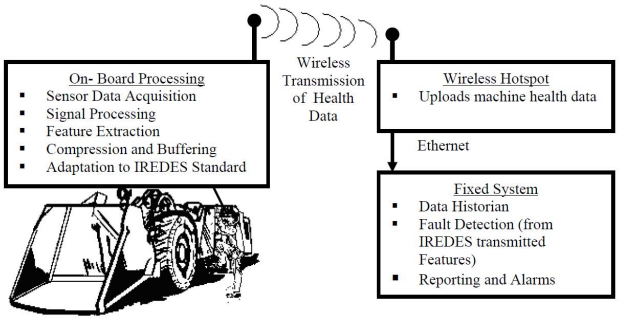

Figure 10 Schematic of mobile equipment monitoring system

B. Use Cases

Computerized condition monitoring systems are as simple as the hand-held vibration monitor and as complex as a dedicated online monitoring system with built in artificial-intelligence techniques. The range reflects the varying cost-benefit and risk analysis for the diversity of equipment in industry; one might not monitor the fan on an HVAC system for personnel refreshment areas but hoists, bottle-neck electro-mechanical shovels, and other operationally-critical equipment might demand the use of a dedicated system.

There is no reason such a condition-monitoring system should not be justifiable for this entire range of scenarios. Where the return does not justify a dedicated monitoring system, the proposed solution should be capable of being "wheeled" from one station to the next at periodic intervals – all the while storing and updating a profile for each one to provide the total functionality described herein. In this fashion, the fully-dedicated system should be almost identical to the sporadic one except in its periodicity.

One individual dedicated-system might monitor a number of machines in close proximity to one another. Such a requirement provides additional value added to the customer for the cost of one fixed system—provided the machines are sufficiently close that wiring will not affect the working environment.

The underground environment is one typically characterized by intermittent and band-limited data networking (similar to other remote monitoring applications as wind-turbine and bridge monitoring). High-level, on-line and in-situ processing of machinery signals should therefore be conducted almost exclusively onboard (initial efforts tend to indicate this as entirely possible despite the experience described in [25] as discussed in section II.C).

The environment in which this system could be deployed is highly variable and likely harsh. Mobile machinery could be subject to heavy vibration and other environmental detractors (such as heat, EMI, and liquids). The hardware must be capable of bearing a wide range of caustic industrial

environments. It may be deployed in remote and difficult to access environments (such as wind turbines or compression stations along a pipeline) or in high traffic areas.

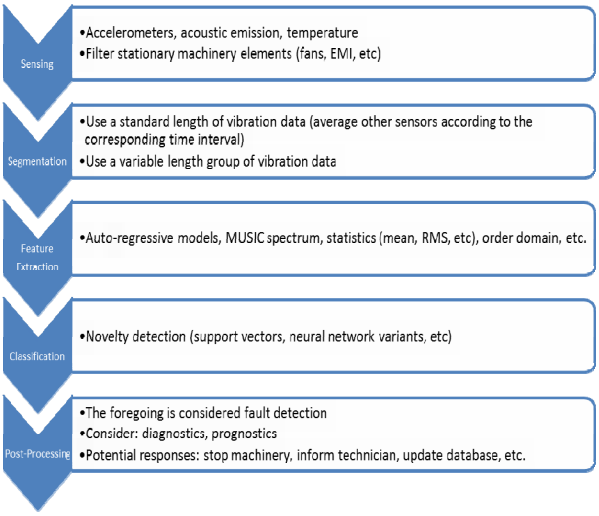

Figure 11 Generic pattern recognition problem (defined by [31]**)**

C. Functionality

After the mechanical interface of the condition monitoring system is configured, the software system should be configurable, maintainable and reviewable from a network location. All aspects of algorithm choice, sub parameters, linkage between algorithms, and reporting choices should be available to the user from anytime from a networked location.

Rudimentary and advanced signal processing techniques should be available such as band-pass/stop filtering, autoregressive models, kalman filtering, resampling (for order tracking), envelope analysis, time-synchronous averaging, etc. Each algorithm has a differing number of configuration parameters (e.g. an AR model might only have one integer parameter identifying the model order while a band-stop filter might need multiple parameters such as stop-band bandwidth and the center blocking frequency). Similarly, each algorithm might have a different number of run-time parameters (e.g. order tracking might require two channels – a tachometer channel and a vibration channel; while AR models might only require one). All algorithm choices could have a similar disparity in configuration and run-time configuration options; this diversity should be supported.

Segmentation routines should range from event-based to simple constant-interval approaches. For instance, a segment might be defined by 15 rotations of a shaft or it could consist of 2 seconds of data. From one segment to another, the former might have a variable amount of data in it while the latter will not (assuming constant sampling rates). This choice has an impact on the quality of classification results – especially in time-varying machines (see [4]). These are options that should be accessible to the user/configurator.

The result from the segmentation stage might feed into a feature-vector generating stage. Feature vectors should be formable from multiple signal and signal-processing combinations; when combining feature vectors, their components should be based on coherent segmentation intervals. A feature vector should also be viewed as a time-varying signal that can be manipulated and combined with other processing and analysis algorithms. Not only should they be available for further processing but they should be available for presentation to network interfaces.

In the authors' work, the use of novelty detection is prominent; while it is a preferred approach in such a highly automated system, other classification algorithms should be available for use. As a matter of implementation, support-vector, auto-associative, and neural-network based algorithms should be available while supporting the future use of more generic pattern recognition and artificial intelligence algorithms. Similar to other algorithms, configuration and run-time parameters may vary in number.

One might regard other post-processing steps such as diagnostics, prognostics and sensor-failure detection as other generic examples of "smart signal processing" not warranting distinction in the actual software design. Like novelty detection, they may process a varying number of channels through a variety of signal-processing algorithms and their outputs may in turn be viewed as data channels for analysis. Further post-processing considerations such as alarm reporting, updating databases and effecting control are application integration issues that will largely be supported by middleware standards like OSA-CBM.

D. Software/Hardware Implementation

In the early stages of the software design cycle, the specification of technologies should admittedly be avoided. This is a difficult principle to observe when building lower level architecture, in practice, due to the limitation of various implementation hardware and software environments. The architecture espoused herein is one that has been designed and validated first in MATLAB object-oriented programming (OOP). MATLAB affords many advantageous in scientific-processing applications that make it a very suitable development environment; while it is used in online embedded systems, the implementation of its (vectorized) code can be inefficient. Nevertheless, it proved to be a highly suitable environment for prototyping work and for the rapid generation of results for research papers.

National Instruments' (NI) embedded hardware solutions and its LabVIEW-enabled automation are viewed as an ideal integrated development environment for the ultimate production of this work. In recent years, LabVIEW has developed significantly – still providing original strengths like its MATLAB-like toolboxes while introducing increasingly advanced software-development options like support for OOP. Its seamless integration with military-specification vibration-rated embedded systems like the CompactRIO is another primary reason for its selection.

In the following section, a generalized object-oriented framework is described that will support the above-described functionality of a smart signal processing system. A deep understanding of LabVIEW and the CompactRIO are not required and the authors will avoid further reference to them in order to make the framework as generic as possible. However, there are important LabVIEW OOP considerations that must be born in mind through this process (e.g. a lack of multiple inheritance).

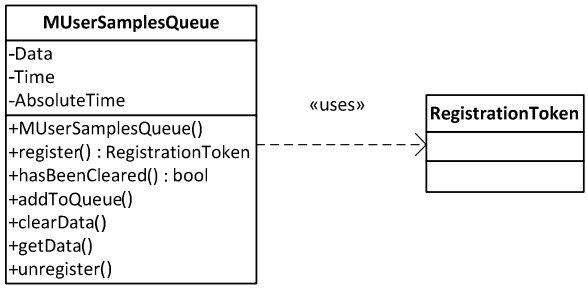

Figure 12 Multiple User Samples Queue (used to allow dynamic registration to data channels; stores an n-ary signal of discrete data that has been sampled at intervals reflected in the time array whose first entry occurred at AbsoluteTime; this class enables the storage of data for registered users that are presumed to retrieve and subsequently relieve the queue of the need to store data for it)

IV. PROPOSED SOFTWARE ARCHITECTURE

The focus of this work is to describe a software architecture that enables extensible "intelligent signal processing." A review of the most critical elements of this architecture will add a set of design patterns to the literature to address this problem class. Design patterns are a generalized reusable solution to common problems in software design; in mechanical/electrical engineering, design patterns are ubiquitous. For instance, the interface specification for AC power outlets allows the engineer to solve the common problem of provision of power to a new system design; this pattern can be further extended to problems of adapting one class of electronics to another (110-V North American power vs. 220-V European power). Design patterns are a cornerstone of modern object-oriented software systems.

While a broad range of potential applications has been identified, the concern herein is fixed on condition-monitoring of time-varying machinery. The architecture is flexible enough to accommodate those other applications but the techniques that are actually implemented are limited to this class. The elements of software design are further limited to the data-processing and analysis layers; no discussion of enterprise architecture is considered here beyond the desire to create interfaces that are compatible with standards like OSA-CBM.

Section III.C and Figure 11 provide an ideal starting point for the system decomposition. A common problem in section III.C is the need for the capacity to dynamically route data. This functionality is supported by the "multiple user samples queue." MUserSamplesQueue's interface is shown in Figure 12; it stores an n-ary signal for subscribed users until they have retrieved the data and indicated they no longer need it to be stored. Only subscribed users are allowed to access data by providing a registration token that is created when they first subscribe to it.

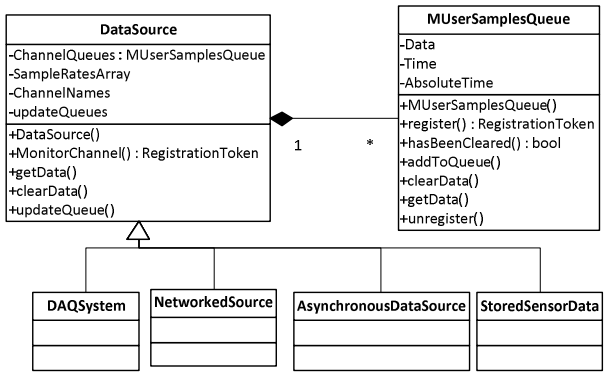

Figure 13 A "DataSource" can be implemented either as a DAQSystem, a NetworkedSource, StoredSensorData, or AsynchronousDataSource; each channel of data that can be provided by a DataSource is implemented using a MUSerSamplesQueue

This multi-user dynamic queue system is employed heavily when implementing data sources such as networked sources, DAQ systems, and stored sensor data. A generic DataSource class is defined in Figure 13 from which these more specific types of data sources take their interface. By employing this approach, once a data source is configured by the user it can be employed by the rest of the system without needing to adapt anything other than the DataSource for the method of data generation. The StoredSensorData class that implements the DataSource interface, for instance, was used during initial architecture validation to simplify software testing by eliminating the need for a live source of data; it was also highly useful in generating a number of research papers generated by the authors in the past few years.

DataSources may be conditioned after segmentation. Figure 14 demonstrates the SignalConditioningStrategy class that describes the generic interfaces for any of a variety of signal conditioning strategies that might be used (e.g. time synchronous averaging, ARMAX, AR, etc.). By specifying a common interface, the system need not know any specifics of the individual strategy after it has been initialized.

A DataSource may have multiple channels – as is the case with a data acquisition system. A DAQ may have multiple hardware cards each with multiple channels onboard. These channels within DataSource are implemented with MUserSamplesQueue; users initially subscribe to a DataSource's channel after which point they can get and clear data from the channel as necessary.

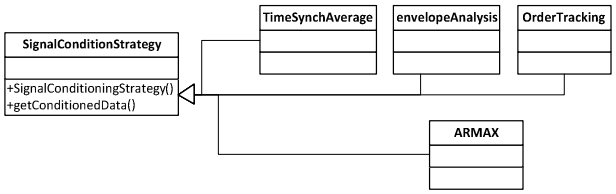

Figure 14 There are a variety of signal conditioning techniques that might be employed (e.g. time synchronous averaging, envelope analysis, order tracking, AR, ARMAX, etc.); they all share the interface of SignalCondtioningStrategy

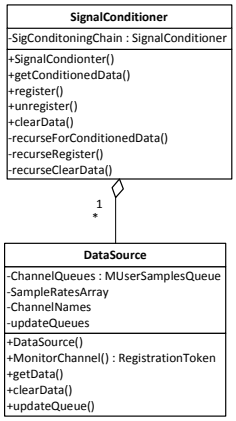

Figure 15 A signal conditioner creates a hierarchy of signal processing and data sources

In order to create a sophisticated and dynamic network of possible signal conditioning options, there is a need for a facility that can route multiple data sources through various signal conditioning strategies. For instance, to generate a signal resampled with order tracking that subsequently smooths the result, there is a need to access multiple data sources and signal conditioners. To accomplish this, the facility must register and access data from a tachometer and at least one accelerometer, map it through an orderTracking conditioner after which the subsequent signal is processed through a filter. At configuration time, the user should be able to specify a large and complex network of signal processing techniques and data sources as suggested by this example. This facility is provided by the SignalConditioner whose interface is shown in Figure 15. SignalConditioner takes a specification consisting of DataSources and SignalConditionStrategies; it dynamically registers itself to each of the DataSources and when called, maps the signals from them through SignalConditioners whose resultant signals may in turn be further processed by SignalConditioners in an inductive fashion.

Feature vectors are ultimately generated by a FeatureGenerator class that gets conditioned signals from a SignalConditioner; these signals are first segmented using a SegmentationStrategy and then reduced to a feature vector by another SignalConditioner. In this approach, the generation of a feature vector is viewed as no different from conditioning a signal since the consequent feature vector is an n-dimensional signal.

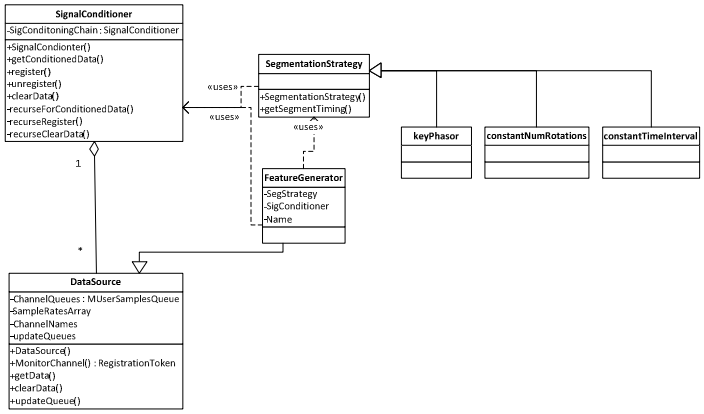

Figure 16 A FeatureGenerator gets conditioned signals from signalConditioners and processes them into feature vectors with its own internal signalConditioner according to data from time intervals specified by a SegmentationStrategy

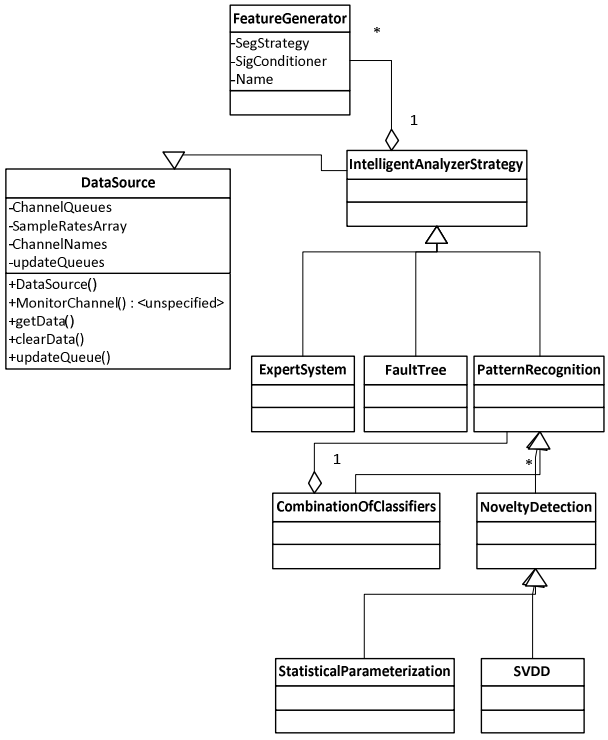

Figure 17 An IntelligentAnalyzerStrategy might be used to perform fault detection, diagnostics, prognostics, sensor-failure analysis, etc.; it is a DataSource and its ouputs may be used in higher-level stages to facilitate all this functionality

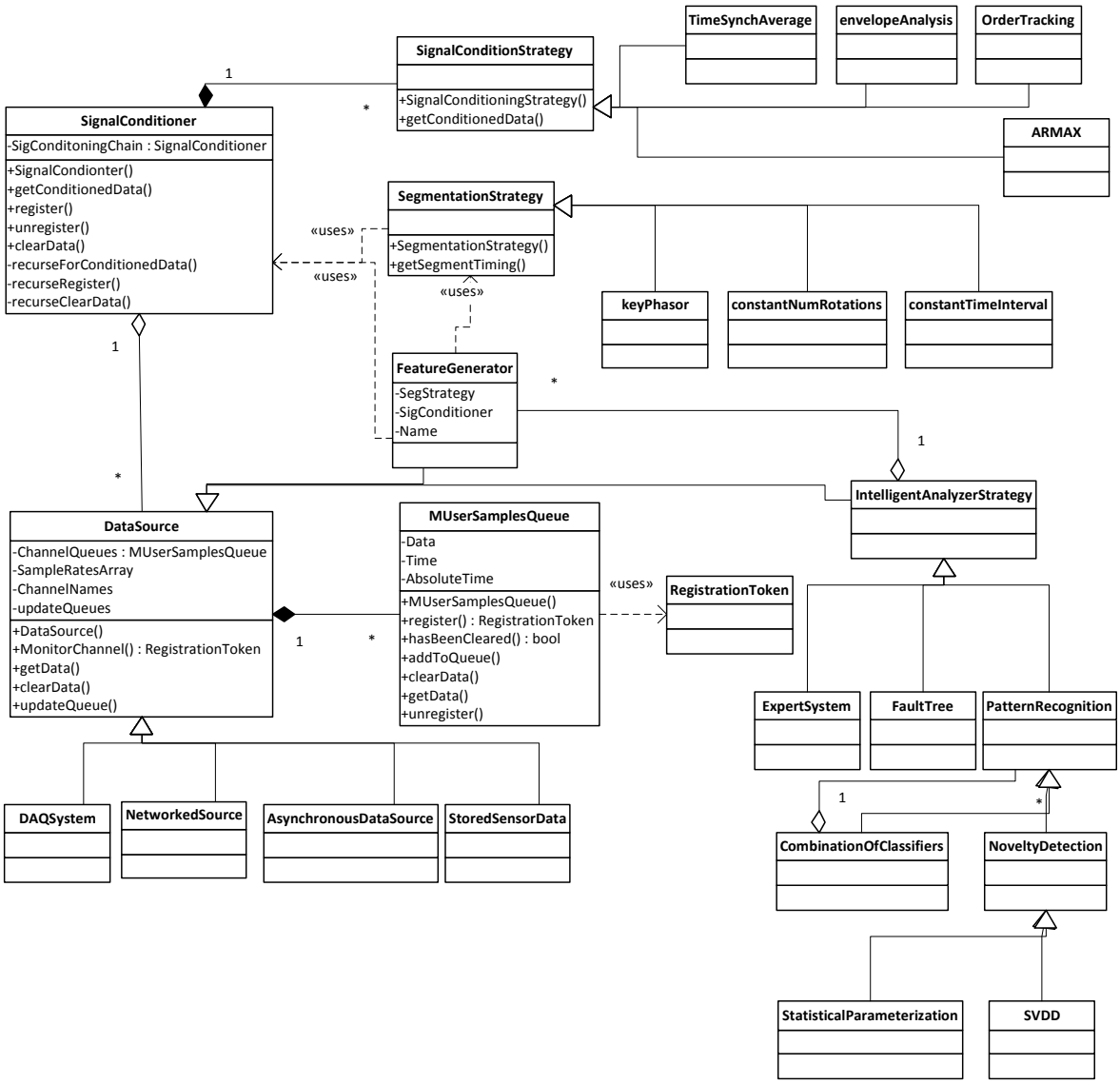

Figure 18 The central elements of the design for an intelligent signal processing system

A variety of segmentation strategies might be employed – such as a fixed interval rule or a rule based on a fixed number of shaft rotations. SegmentationStrategy provides a list of time intervals that are used to generate a feature vector. A FeatureGenerator is a DataSource that may store feature vectors and that may be queried by higher level users. After processing, the array of n-dimensional feature vectors may be regarded as an n-dimensional signal – potentially an asynchronous one depending on the employed segmentation strategy. Users of a FeatureGenerator may access it in precisely the same manner they do a DataSource. The interactions of FeatureGenerator, SegmentationStrategy, SignalConditioner and DataSource are shown in Figure 16.

Finally, the results from the feature-generation stage are employed by an IntelligentAnalyzerStrategy; this class permits the broad range of applications described in Figure 1. In this work the focus is on the detection of faults and their diagnosis and prognosis as well as sensor-failure analysis. Figure 17 demonstrates the interactions involved with the IntelligentAnalyzerStrategy; pattern recognition is a prominent element in this decomposition. The work described in section II is enabled in this design by the inclusion of the employed novelty-detection techniques that are children of the PatternRecognitionClass in Figure 17.

Further design is required to permit the specification of the graphical user interface (GUI) and integration with enterprise-level applications. What has been described

should be sufficiently extensible to easily facilitate this task and has been demonstrated, through the author's experience as described in section II, to provide a flexible means of solving the intelligent signal processing problem of fault detection in variable state machinery. The design should also enable future work and software additions including diagnosis, prognosis, fault detection and more generic intelligent signal processing challenges. The complete design is shown in Figure 18.

V. IREDES AUGMENTATION FOR CONDITION MONITORING

This work has explored the problem of condition-monitoring of variable speed and load machinery including applications with mobile underground mining equipment. The solution to this problem in the industrial environment is further embellished with existence of an excellent enterprise-level integration standard like OSA-CBM. Therefore, when the IREDES endeavor is prepared to expand its standard for condition-monitoring applications, its developers may choose to start with OSA-CBM and some of the considerations contained herein.

Moreover, the IREDES endeavor should review MIMOSA's Open Systems Architecture for Enterprise Application Integration (OSA-EAI) and consider integrating the IREDES standard within this framework. OSA-EAI defines "data structures for storing and moving collective information about all aspects of equipment, including platform health and future capability into enterprise applications."[22] Both these standards have wide industrial support outside the mining sector; MIMOSA participants like Caterpillar and Rockwell Automation Systems should provide further motivation for IREDES to consider these standards as they could ease the adoption of IREDES in the future.

VI. CONCLUSION

There is a broad range of problems that require signal processing and intelligent signal analysis. Condition monitoring of equipment subject to stationary conditions and time-varying conditions like mobile underground mining equipment is one such critical problem. An architecture that considers this broad range of problems, one that can address the specific problem of CBM, was developed; it permits dynamic run-time configuration of signals through signal processing algorithms – the results of which may be analyzed by intelligent algorithms. The design proposed within should permit integration with OSA-CBM and consequently will enable the integration of the information developed with enterprise-level applications.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of the Center of Excellence in Mining (CEMI) in Sudbury, Ontario and Vale for their financial and in-kind support. The authors are also appreciative of Dr. Chris Mechefske's support through the loan of an experimental gearbox. Finally, the authors are grateful for the efforts of Jim Provost for his service in reviewing this work.

REFERENCES

[1] M Markou, S Singh. Novelty detection: A review - Part 1: Statistical approaches, Signal Processing. 83 (2003) 2481-2497.
[2] M Markou, S Singh. Novelty detection: A review - Part 2: Neural network based approaches, Signal Processing. 83 (2003) 2499-2521.
[3] H Sohn, K Worden, CR Farrar, Novelty detection under changing environmental conditions, Proceedings of SPIE - The International Society for Optical Engineering. 4330 (2001) 108-118.
[4] J McBain, M Timusk. Fault detection in variable speed machinery: Statistical parameterization, Journal of Sound and Vibration. 327 (2009) 623-646.
[5] M Timusk, M Lipsett, CK Mechefske. Fault detection using transient machine signals, Mechanical Systems and Signal Processing. 22 (2008) 1724.
[6] MA Timusk, MG Lipsett, CK Mechefske, Automated duty cycle classification for online monitoring systems, 1 PART A (2008) 563-573.
[7] MA Timusk, CK Mechefske, M Lipsett. A pragmatic framework for on-line neural network based detection of machinery, International Journal of COMADEM. 8 (2005) 35-41.
[8] J McBain, M Timusk. Novelty Detection Augmented for Fault Detection in Variable Speed Machinery, Journal of Vibration and Acoustics. (2012).
[9] J McBain, M Timusk. Feature extraction for novelty detection as applied to fault detection in machinery, Pattern Recognition Letters. 32 (2011) 1054-1061.
[10] J Haiwei, Y Haitao, L Dong, Gearbox Diagnosis Based on N4SID Subspace Method, 2 (2010) 383-6.
[11] DMJ Tax, DDtools, the Data Description Toolbox for Matlab, 1.7.0 (2008).
[12] M Timusk, D Crymble, J McBain, An integrated flexible platform for development of fault detection systems and duty cycle simulation, (2010) 373-376.
[13] G Chen, IREDES On-line: Online Data Exchange of Automation Systems for the Mining Industry, (2008).
[14] C Groba, S Cech, F Rosenthal, A Gossling, Architecture of a predictive maintenance framework, (2007) 59-64.
[15] M Rao, J Zhou, H Yang, Architecture of integrated distributed intelligent multimedia system for on-line real-time process monitoring, 2 (1998) 1411-1416.
[16] C Emmanouilidis, E Jantunen, J MacIntyre. Flexible software for condition monitoring, incorporating novelty detection and diagnostics, Computers in Industry. 57 (2006) 516-27.
[17] P Hannah, A Starr, A Ball, Decisions in condition monitoring-an examplar for data fusion architecture, 1 (2000) 5-23.

[18] Y Li, L Chun, ANY Ching, An agent-based platform for Web-enabled equipment predictive maintenance, (2005) 132-5.
[19] T Pirttioja, A Pakonen, I Seilonen, A Halme, K Koskinen, Multi-agent based information access services for condition monitoring in process automation, 2005 (2005) 240-245.
[20] T Jackson, M Fletcher, B Liang, M Jessop, J Austin, An architecture for distributed search and data-mining in condition monitoring applications, (2007) 1-12.
[21] L Zhang, A Mathew, S Zhang, M Lin, Implementation of asset health assessment system with pattern-oriented design and practice, (2006) 666-671.
[22] KD Bever, Understanding MIMOSA'S open standards for on-board health assessment and enterprise application integration, 3 (2007) 2110-2123.
[23] MIMOSA, OSA-CBM UML Specification 3.3.1, (2010).
[24] B Chidambaram, DG Gilbertson, K Keller, Condition-based monitoring of an electro-hydraulic system using open software architectures, (2005) 3532-9.
[25] J Dunsdon, M Harrington, The application of open system architecture for condition based maintenance to complete IVHM, (2008) 1-9.
[26] K Keller, D Wiegand, K Swearingen, C Reisig, S Black, A Gillis, et al., An architecture to implement Integrated Vehicle Health Management systems, (2001) 2-15.
[27] XB He, JC Ning, A multi-agent architecture of health monitoring system for long-span bridge, (2009) Communication University of China; Wuhan University; James Madison University; Institute of Policy and Management, Chinese Academy of Sciences; IEEE Wuhan Section.
[28] Min-Chun Pan, Po-Ching Li, Yong-Ren Cheng. Remote online machine condition monitoring system, Measurement. 41 (2008) 912-21.
[29] I Seilonen, A Tuomi, J Olli, K Koskinen, Service-Oriented Application Integration for condition-based maintenance with OPC Unified Architecture, (2011) 45-50.
[30] F Zhao, J Chen, G Dong, L Guo. SOA-based remote condition monitoring and fault diagnosis system, International Journal of Advanced Manufacturing Technology. 46 (2010) 1191-1200.
[31] RO Duda, PE Hart, DG Stork, Pattern Classification, 2nd Edition ed., Wiley, New York, NY, 2001.

Jordan J. McBain is a Ph.D. candidate in Natural Resources Engineering at Laurentian University in Sudbury, Ontario Canada. His undergraduate degree in Software Engineering and Management was completed in 2005 at McMaster University in Hamilton, Ontario Canada.

He was enrolled as a Naval Combat Systems Engineering Officer (NCSEO) in the Royal Canadian Navy's (RCN) Regular Officer Training Plan during the course of his undergraduate work. After graduating and on completing his technical training at the Canadian Forces Naval Engineering School and his practical training onboard Her Majesty's Canadian Ship REGINA, he was posted into a project management role as project control officer in a major capital Crown project. He released from the RCN in 2008 at which point he took up MASc studies at Laurentian (and subsequently transferred into the Ph.D. program without completing a masters). In 2009, he suspended his Ph.D. studies to take a post-doctorate position with Liten's Automotive to complete the development of a torsional vibration simulator's control and diagnostics system. His present research interests include fault detection, diagnosis, and prognosis of machinery – particularly machinery subject to non-stationary conditions.

Mr. McBain is a licensed Professional Engineer with the Professional Engineers of Ontario.

Markus A. Timusk completed a Ph.D. in Mechanical Engineering at Queen's University in Kingston, Ontario Canada in 2006. His Masters of Engineering Science and Bachelors of Engineering Science were both completed at the University of Western Ontario, in London, Ontario Canada in 2001 and 1999 respectively.

Dr. Timusk is the founding program coordinator and an assistant professor in the mechanical engineering program at the Bharti School of Engineering at Laurentian University in Sudbury Ontario Canada. He is the principal investigator with a number of projects sponsored by such partners as Liten's Automotive, the Center of Excellence in Mining Innovation, Vale Inco, Rail-Veyor Material Haulage, etc. His research interests focus on the development of automated, artificial-intelligence based fault detection systems and supporting subject matter like mechatronics, pattern recognition, machine measurement, vibration analysis and signal processing. His engineer-in-training work with Litens Automotive in developing automotive components and reliability testing facilities lead naturally to his graduate work in condition monitoring. His Masters was sponsored by Syncrude Canada and focused development of fault detection systems for mining equipment. Similarly, his Ph.D. was further sponsored by Syncrude but was focused on fault detection in electro-mechanical shovels.

Dr. Timusk is a licensed Professional Engineer with the Professional Engineers of Ontario.