Notice of Violation of IEEE Publication Principles

"Agents and Multi-agents Systems and Application to Condition Monitoring"
by Christina B. Vilakazi, Tshilidzi Marwala
in the Proceedings of the 2007 International Conference on Systems, Man and Cybernetics
2007, pp. 644-649

After careful and considered review of the content and authorship of this paper by a duly constituted expert committee, this paper has been found to be in violation of IEEE's Publication Principles.

This paper contains significant methodology and text from the paper cited below. The content was used with insufficient attribution (including appropriate references to the original author(s) and/or paper title) and without permission.

"The Design of a Multi-Agent Transformer Condition Monitoring System"
by Stephen D.J. McArthur, Scott M. Strachan, and Gordon Jahn
in the IEEE Transactions on Power Systems, Vol 19, No 4, November 2004, pp. 1845-1852

Agents and Multi-agent Systems and Application to Condition Monitoring

Christina B. Vilakazi and Tshilidzi Marwala

Abstract— **In recent years the need for improved monitoring systems has necessitated the application of intelligent systems. In this study, a multi-agent system (MAS) is used for vibration condition monitoring. The system consists of functional agents such as feature extraction agent, interpretation agent, diagnosis agent, engineering assistant agent and registry agent. Support Vector Machine (SVM), Multi-layer perceptron (MLP) and Extension Neural Network (ENN) are used as interpretation agent. The rationale behind using multi-agent system for condition monitoring is explained. Experimental results show that multi-agent system is an effective artificial intelligence techniques for condition monitoring.**

I. Introduction

Condition monitoring of machines is gaining importance in industry due to the need to increase machine reliability and decrease the possible loss of production caused by machine breakdown. By definition, condition monitoring is performed when it is necessary to access the state of a machine and to determine whether it is malfunctioning through reason and observation [1]. Condition monitoring has become increasingly important due to an increase need for normal undisturbed operation of equipment in industry. An unexpected fault or shutdown can result in a serious accident and financial loss for the company. Hence, companies must find ways to avoid failures, minimize downtime, reduce maintenance costs, and lengthen the lifetime of their equipment.

One of the most common components in modern rotating machinery is the rolling element bearing. Most machine failures are linked to bearing failure [2], which often result in lengthy downtime that have economic consequences. Hence, a reliable, fast and automated diagnostic technique allowing relatively unskilled operators to make important decisions without the need for a condition monitoring specialist to examine data and diagnose problems is required. The most commonly used condition monitoring system is vibration-based condition monitoring. Vibration monitoring is based on the principle that all systems produce vibration. When a machine is operating properly, vibration is small and constant; however, when faults develop and some of the dynamic processes in the machine change, vibration spectrum also changes [3]. Various artificial intelligence techniques have been applied for vibration-based condition monitoring. During the last decade artificial neural network based models such as multilayer perceptron (MLP) and radial basis function (RBF) have been used extensively for bearing condition monitoring [4][5]. Lately, kernel-based classifiers such as Support Vector Machine (SVM) have been used for bearing fault diagnosis [6] [7]. Data-based statistical approaches have achieved considerable success in speech recognition and have been recently used for condition monitoring [8][9].

After studying various condition monitoring system, a number of requirements for an effective condition monitoring system were identified. Firstly, the system must be able to capture and condition the relevant data automatically. Secondly, it should be able to learn trends and behavioral characteristics of each individual data points, and inter-parameter relationships, over time and different operational conditions. Thirdly, it should have automatic interpretation of the conditioned data for identifying incipient defects and provide clear and concise defect information and remedial advice to the operation engineer. Lastly, the system should be extensible so as to include further interpretation technology and monitoring technologies such as partial discharge monitoring and vibration monitoring. Multi-agent system was identified as a techniques that can be used to meet the aforementioned requirements. This paper will outline the development of a vibration-based condition monitoring system using multi-agent. Intelligent agents were encapsulated with different Artificial Intelligence (AI) paradigms such as SVM, extension neural network(ENN) and MLP for bearing condition monitoring.

II. Agents and Multi-Agent System

Agents are software systems that function autonomously to achieve desired objectives in their environment. Agents must exhibit four characteristics: autonomy, social ability, pro-activeness and reactiveness. Autonomy means each agent will operate in an independent mode, continually performing its function while altering its behavior as required. Social ability implies each agent can cooperate and communicate with other agents. Reactivity and proactiveness means that the agents have the ability to solve problems and ensure that they deliver the correct information or initiate the required control activity. A multi-agent system (MAS) is defined as a loosely coupled network of problem solvers that work together to solve problems that are beyond the individual capabilities or knowledge of each problem solver [10]. The increasing interest in MAS research is due to advantages inherent in such systems, including their ability to solve problems that may be too large for a centralized single

Christina B. Vilakazi is with the School of Electrical and Information Engineering, University of the Witwatersrand, Private Bag 3, Wits, 2050 t.marwala@ee.wits.ac.za

T. Marwala is the Professor of Electrical Engineering at the School of Electrical and Information Engineering, University of the Witwatersrand, Private Bag 3, Wits, 2050 t.marwala@ee.wits.ac.za

1-4244-0991-8/07/$25.00/©2007 IEEE

agent, provide enhanced speed and reliability and tolerance for uncertainty in data and knowledge.

There are several motivations for using a multi-agent system. Firstly, they are able to solve problems that are complex for a centralized single agent to do due to resource limitations or the sheer risk of having one centralized system. Secondly, to enhance modularity, which reduces complexity, speed due to parallelism, reliability due to redundancy, flexibility (i.e. new tasks are composed more easily from the more modular organization) and reusability at the knowledge level hence shareability of resources. Lastly, MAS offer the extensibility and flexibility framework for integrating the necessary data capture system, monitoring system and interpretation function [10]. Based on the functional requirement, MAS was seen as an essential technology for vibration-based condition monitoring system. This technology permits the development of more intelligent and automated diagnostic and monitoring functions.

III. Potential of MAS

Multi-agent systems have proven to be an effective paradigm in a number of distributed networked applications that require information integration from multiple heterogeneous autonomous entities. More recently, MAS have begun to emerge as an integrated solution approach to distributed computing, communication, and data integration needs in industry.

A. Agent-based Computing and Agent-oriented Programming

A multi-agent system consisting of multiple agents can take advantage of computational resources and capabilities that are distributed across a network of interconnected entities. An agent-based approach allows for the creation of systems that are flexible, robust, and can adapt to the environment [10]. This is especially helpful when components of the system are not known in advance, change over time, and are highly heterogeneous. Agent-based computing offers the ability to decentralize computing solutions by incorporating autonomy and intelligence into cooperative, distributed applications. Each agent perceives the state of its environment, infers by updating its internal knowledge according to the newly received perceptions, decides on an action, and acts to change the state of the environment. Agent-oriented programming is a software paradigm used to facilitate agent-based computing and extends from object-oriented programming by replacing the notions of class and inheritance with the notions of roles and messages, respectively [10].

B. Knowledge-level Communication Capability

Within a multi-agent system, agents can communicate with each other using agent communication languages, which resemble human-like speech actions more than typical symbol-level program-to-program communication protocols [10]. This capability enables agents to distill useful knowledge from voluminous heterogeneous information sources and communicate with each other on the basis of which they coordinate their actions. By enabling performance of computation where computing resources and data are located, and allowing for flexible communication of relevant results to relevant entities as needed, MAS offer significant new capabilities to power systems, which have for so long depended on various forms of expensive telemetry to satisfy most communication needs.

C. Distributed Data Access and Processing

Another benefit offered by MAS is the distribution of agents across a network [10]. Software agents may have different levels of intelligence, ranging from data agents and functional agents to decision agents. Because each agent is designed to perform a specific role, with associated knowledge and skills, distributed and heterogeneous information may be efficiently assimilated locally and utilized in a coordinated fashion in distributed knowledge networks, resulting in reduced information processing time and network bandwidth in comparison to that of more traditional centralized schemes.

D. Distributed Decision Support

MAS also offers a powerful task decomposition approach to problem solving through interaction among agents. This is facilitated by the ability of different agents to coordinate behavior through cooperation by agents establishing mutually agreeable objectives, negotiation by agents negotiating until agreement is reached, or mediation by agents resolving conflicts that cannot be resolved by appealing to a third, neutral agent [11].

IV. Multi-agent System Methodology

Several MAS paradigms and methodologies have been proposed in the literature such MASSIVE [11], DESIRE [12], Gaia [13] and MaSE [14], based on different notions of agents and multi-agent organizations. In this paper, the design framework as described in [15] was followed in designing a multi-agent system for bushing condition monitoring. Figure 1 shows the architecture used for developing the multi-agent system. The three main phases, as depicted in the figure are; requirements capture and task decomposition; agent modelling and agent interactions modelling.

A. Requirements Capture and Task Decomposition

This is the first stage which identifies the application domain, overall problem, objectives, MAS application environment such as information that will be available to an agent, actions required of the agents, and operational and performance constraints. In this stage, task decomposition is performed to determine what the system is supposed to do and not how it is supposed to do it to achieve overall MAS objectives. In the agent community; decomposition is concerned with the partitioning of the problem domain into agents. The main focus of this phase is gaining an understanding of what the system does in the abstract, which serves as a starting point for the architecture development

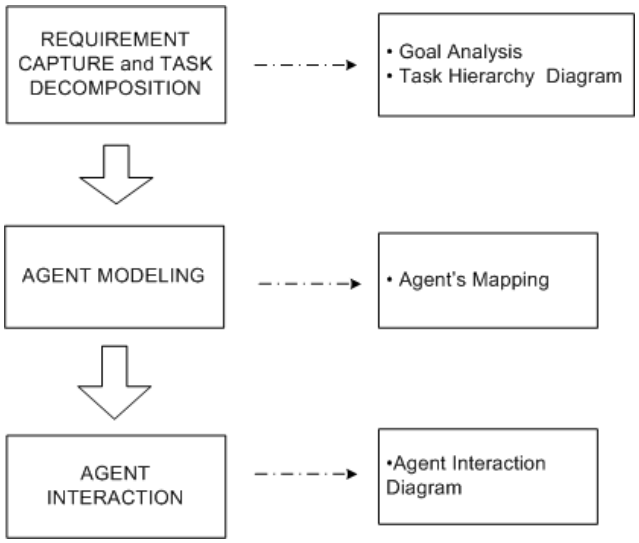

Fig. 1. Architecture development for the multi-agent system

process. The activities undertaken in this phase pertain to understanding the application domain, identifying goals and boundaries of the system, and relating them to the agent design.

B. Agent Modelling

Having decomposed the problem into constituent tasks, the next stage is to identify the agents required to effectively perform the tasks in terms of definition of agent roles; identifying the types of interactions needed between different agents in order to achieve individual or joint goals; and specifying the organization of the different agents in terms of a society of agents that is consistent with the various defined roles and that achieves the overall objectives. Figure 2 depicts the bearing condition monitoring system with different agents.

1) Feature Extraction Agent: Feature extraction agent was designed to provide the necessary data to the interpretation agents. Many parameters may be used to characterize vibration signal. Basic statistical parameters are used to form the feature vector, providing some indication of the shape (cross-correlation), symmetry (skew), peaked ness (kurtosis) of the vibration signal, effectively producing the fingerprint of the vibration signal. The feature vector consists of parameter capable of differentiation between different faults. In this work, one feature extration agent is used since decided to extract the statistical features only.

2) Interpretation Agents:

a) Extension Neural Network Agent: ENN is a new pattern classification system based on concepts from neural networks and extension theory [16]. The extension theory uses a novel distance measurement for classification processes. The input layer nodes receive an input feature pattern and use a set of weighted parameters to generate an image of the input pattern. There are two connection weights between input nodes and output nodes; one connection represents the lower

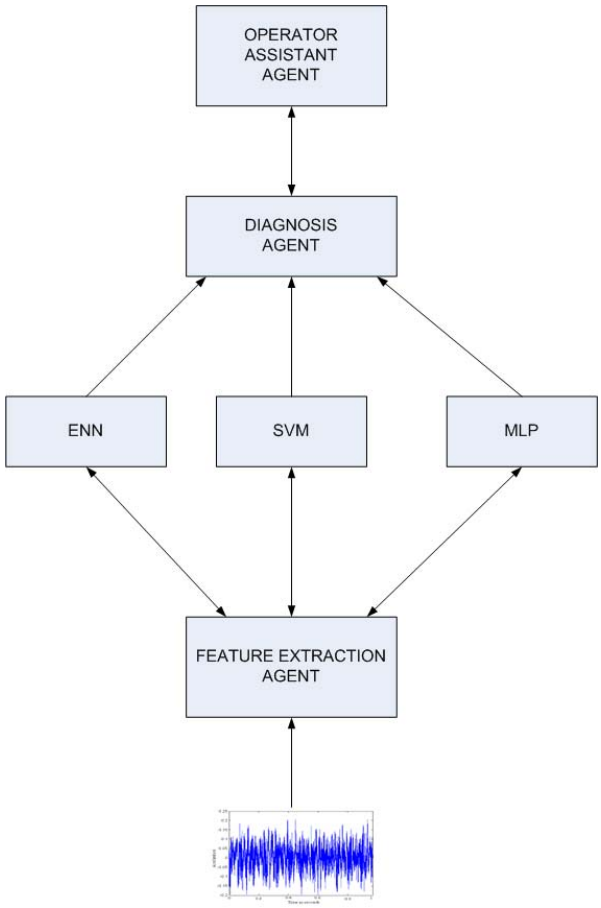

Fig. 2. Bearing condition monitoring system

bound for this classical domain of the features and the other represents the upper bound. ENN uses supervised learning, which tune the weights of the ENN to achieve a good clustering performance or to minimize the clustering error. The network is trained by adjusting the network weights and recalculating the network centres for each training pattern depending on the extension distance (ED) of that pattern to its labelled cluster. The important parameter that has to be selected is the learning rate, as it affects the performance of the network. A detailed description of the ENN can be found in [16].

b) Kernel-Based Agent: Kernel-based classifiers have recently been used as popular and powerful tools for classification, due to their strong theoretical origin from statistical learning theory as well as their high performance in practical applications [17]. SVM classifiers are kernel-based learning algorithms, determining the optimal hyperplane decision boundary in the feature space. In kernel-based algorithms, a kernel transformation process the data in a feature space without the explicit knowledge of a non-linear mapping from the data space to a feature space. In statistical learning theory, the complexity term of the upper bound of the expected risk is minimized by maximizing the margin of

the separating hyperplane. The minimization of the upper bound can be viewed as avoiding the over-fitting problem. The maximization of the margin can be formulated as a quadratic optimization problem so that a global solution can be easily obtained. A detailed description of SVM is found in [17].

c) Multi-layer Perceptron Agent: MLP employs supervised learning in the training of a neural network. The input data vector is presented to the network input layer while the output layer is presented with the "target" output. The network is refined through a process of error backpropagation, where the resultant error between the actual and target output is minimized. A detailed description of the multi-layer perceptrons can be found in [18].

3) Diagnosis Agent: This agent takes the output of the interpretation agents and composes an overall diagnostic conclusion. It employs the confidence provided by the interpretation agents to determine an overall diagnosis and confidence. The program uses weighted majority voting to combine the output of the various interpretation agent. This means that the interpretation layers will be combined according to their performance. The results of the weighted majority voting scheme are computed as follows; initially, a table is built with technique tabulated against conclusion. The cell data is then populated with the various results that have been submitted to the diagnosis agent. After this each time new results are added, the results are recalculated. After the table is fully populated, the weighted majority voting is used to determine an outcome of the events.

4) Engineer Assistant Agent: The engineer assistant agent is designed to present information to the relevant engineer. This is designed to handle diagnostic information from a number of different machine for which the operator is responsible for.

5) Registry agent: The registry agent maintains information about the published variables and monitored conditions for each agent. It is required that all agents must register with the registry agent. The registry agent maintains the current status of all the registered agents. Agent status is a combination of two parameters alive and reachable. The status of a communication link between any two agents is determined by attempting to achieve a reliable communication between them. The registry agent is also used to find information about agents who may supply required data.

C. Agent Interaction

Agent interaction defines communications and exchange of information between different agents. Agents in the condition monitoring system communicate with each other by sending messages in the Agent Communication Language specified by Foundation for Intelligent Physical Agents (FIPA). Agent interaction is facilitated by the registry agent using subscribes, query, inform and confirm command. These messages correspond to registration requests, information requests and other agent actions. A brief overview of agent actions and the corresponding messages is given below.

Registration of an agent with the registry agent: Each agent must register with the registry agent. A registration message includes the registering agent's agent-id, list of published variables. The registry agent issues a confirmation message upon successfully entering the new agent in its database.

Information request by an agent about other agents: To find agents capable of providing required input data, the agent sends a search request to the agent registry. The search request includes the requester's agent-id, and the required variables.

Registry agent's reply to an information request: Upon receiving a search request, the registry agent verifies that the request is legitimate before searching its database to determine which agents can supply the requested variables and the status of these agents. The message from the registry agent to the requester includes the requested variable name, the agent-id of the agent publishing the variable and the status of the requested agent.

Request for belief subscription: Upon receiving the list of agents capable of providing the required input from the registry, the subscribing agent sends requests directly to these agents. A subscription request consists of the requester's agent-id, requested input variable name, the duration of subscription time, the desired time interval between subsequent updates, and a request-id.

Belief-update messages: Upon receiving a belief subscription request the publishing agent sends regular updates within the agreed intervals and duration of the subscription. The message contains the request-id, the sender's id and the requested information.

Figure 3 depicts a sample agent interaction for the condition monitoring system.

V. Experimentation

Having designed condition monitoring system using the most appropriate features of existing MAS design methodologies, the next stage was to implement the prototype. To achieve this the multiagent building toolkit, JADE (Java Agent DEvelopment Framework) was used. JADE is a middle-ware that could be used to develop agent-based applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems. JADE is java-based and provides the infrastructure for agent communication in distributed environments, based on FIPA standards.

A. Vibration Data

The investigation in this study is based on the data obtained from Case Western Reserve University website. The experimental setup comprised of a Reliance Electric 2HP IQPreAlert connected to a dynamometer. Faults of size 0.007, 0.014, 0.021 and 0.028 inches were introduced into the drive-end bearing of a motor using the Electric Discharge Machining (EDM) method. These faults were introduced separately at the inner raceway, rolling element and outer raceway. An impulsive force was applied to the motor shaft and the resulting vibration was measured using two accelerometers, one mounted on the motor housing and

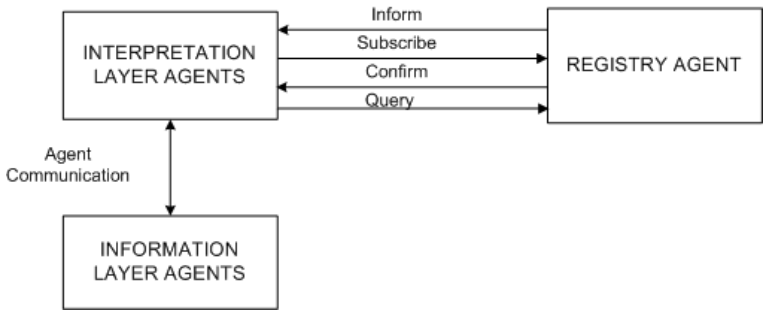

Fig. 3. Sample agent interaction

the other on the outer race of the drive-end bearing. All signals were recorded at a sampling frequency of 12 kHz.

B. Experimental Results

An MLP that consists of 7 inputs layer nodes, 10 hidden layer nodes and 4 output layer nodes was used. Table I shows the normalized confusion matrix resulting from the MLP agent. The backpropagation neural network gave an overall classification accuracy of 100%, which is used as the voting weight for an MLP in the diagnosis stage.

TABLE I
NORMALIZED CONFUSION MATRIX FOR MLP

	Normal	Inner	Outer	Ball
Normal	100	0	0	0
Inner	0	100	0	0
Outer	0	0	100	0
Ball	0	0.02	0	99.8

A radial basis function kernel function was used to train the support vector machines. The normalized confusion matrix for the SVM is shown in Table II. The overall classification accuracy of the SVM is 100%, which is the voting weight for SVM.

TABLE II
NORMALIZED CONFUSION MATRIX FOR SVM

	Normal	Inner	Outer	Ball
Normal	100	0	0	0
Inner	0	100	0	0
Outer	0	0	100	0
Ball	0	0	0	100

The learning rate of ENN was chosen to be 0.534, the values was selected empirically. Table III shows the normalized matrix for the ENN. ENN gave an a classification accuracy of 99.98%. All the interpretation agents gave very good classification accuracy, this might be due to the fact that the feature extracted gave a good signature of the different bearing conditions.

TABLE III
NORMALIZED CONFUSION MATRIX FOR ENN

	Normal	Inner	Outer	Ball
Normal	100	0	0	0
Inner	0	100	0	0
Outer	0	0.02	99.8	0
Ball	0	0.05	0	99.5

For experimental purposes, one data instance is passed to the data preprocessing agent and processed as described previously. The preprocessed data is now sent, via agent subscription and messaging, to each of the interpretation layer agents. The interpretation layer agents work on data simultaneously and the agent will forward their results to the diagnosis agent in order of completion. From these, the first conclusions are generated by the multi-layer perceptron agent and this passes the result to the diagnosis agent using the agent interactions described. The agent returns results detailing the confidence or probability due to one of the possible condition. The diagnosis table is as shown in Table IV. The table shows that the bearing is operating under normal condition with a confidence level of 100%. Next, the

TABLE IV
COMBINED RESULTS FOR THE BACKPROPAGATION NEURAL NETWORK

	Normal	Inner	Outer	Ball
MLP	1	0	0	0

kernel-based agent provides its result. This yields the table shown in Table V shows that the bearing is operating under normal conditions with a confidence level of 100%. The last

TABLE V
COMBINED RESULTS FOR THE BACKPROPAGATION NEURAL NETWORK AND KERNEL-BASED CLASSIFIER

	Normal	Inner	Outer	Ball
MLP	1.0000	0	0	0
SVM	1.0000	0	0	0
Results	1.0000	0	0	0

result passed to the diagnosis agent is the conclusion from the extension neural network. When added to the table, the result is shown in Table VI. At this stage, the bearing diagnosis agent concludes its calculation of the combined result. It determines that bearing is operating under normal conditions with a confidence level of 100%. This is now passed to the Engineer Assistant Agent for display. The system actually predicted the condition correctly as the normal operation condition was passed to the system.

TABLE VI

COMBINED RESULTS FOR THE BACKPROPAGATION NEURAL NETWORK AND KERNEL-BASED CLASSIFIER AND EXTENSION NEURAL NETWORK

	Normal	Inner	Outer	Ball
MLP	1.0000	0	0	0
SVM	1.0000	0	0	0
ENN	0.9998	0	0	0
Results	0.9999	0	0	0

VI. DISCUSSION AND CONCLUSION

This paper demonstrated a design of a condition monitoring system using intelligent agents and multi-agent systems. A conceptual framework for condition monitoring was designed based on the requirements of condition monitoring. A functional design of the condition monitoring consists of four layers; the data monitoring layer, interpretation layer, diagnostic layer and information layer, which were presented. The data monitoring layer consists of two agents which are responsible for extracting features and conditioning of data received from the measurement system. The interpretation layer consists of three agents, the kernel-based agent, backpropagation agent and extension neural network agent. The interpretation agents are encapsulated with classifiers hence, were trained prior to usage. These agents interpret data from the data monitoring layer and give a diagnosis on the condition of the equipment. The diagnosis layer uses weighted majority voting to combine the decision from the interpretation agents. This information along with the maintenance recommendations are passed to the operator assistant agent.

The adoption of a distributed MAS architecture also provides a basis for continual improvement of engineering decision support due to the flexibility and scalability achieved by the use of common communication mechanism. This system will enable integration of further data sources and interpretation agents.

ACKNOWLEDGEMENTS

The financial assistance of the National Research Foundation (NRF) of South Africa, the Carl and Emily Fuchs Foundation and the Council of Scientific and Industrial Research towards this research is hereby acknowledged. Opinions and conclusions arrived at, are those of the authors and are not necessarily to be attributed to the NRF.

REFERENCES

[1] J. H. Williams, A. Davies, and P. R. Drake, *Condition-Based Maintenance and Machine Diagnostics*. Chapman and Hall, 1992.

[2] X. Lou and K. Loparo, "Bearing fault diagnosis based on wavelet transform and fuzzy inference," *Mechanical Systems and Signal Processing*, vol. 18, pp. 1077–1095, 2004.

[3] T. Marwala, *Fault identification using neural network and vibration data*, ch. Doctoral Thesis. University of Cambridge, 2001.

[4] B. Samata, "Gear fault detection using artificial neural network and vector machines with genetic algorithms," *Mechanical Systems and Signal Processing*, vol. 18, pp. 625 – 644, 2005.

[5] B. Yang, T. Han, and J. An, "ART-KOHONEN neural network for fault diagnosis of rotating machinery," *Mechanical Systems and Signal Processing*, vol. 18, pp. 645 – 657, 2004.

[6] A. Rojas and A. Nandi, "Practical scheme for fast detection and classification of rolling-element bearing faults using support vector machines," *Mechanical Systems and Signal Processing*, 2006.

[7] B. Samanta and K. R. Al-Bushi, "Artificial neural network based fault diagnostic of rolling elements bearing using time-domain features," *Mechanical Systems and Signal Processing*, vol. 17, pp. 317–328, 2003.

[8] H. M. Ertunc, K. Loparo, and H. Ocak, "Tool wear condition monitoring in drilling operations using hidden markov models," *International Journal of Machine Tools and Manufacture*, vol. 41, pp. 1363 – 1384, 2001.

[9] T. Marwala, U. Mahola, and F. Nelwamondo, "Hidden Markov models and gaussian mixture models for bearing fault detection using fractals," *Proceedings of the International Joint Conference on Neural Network*, pp. 1535 – 1571, 2006.

[10] M. Wooldridge, *An introduction to multiagent system*. England: John Wiley and Sons, 2001.

[11] J. Lind, "Iterative software engineering for multiagent systems the MASSIVE method," *in Lecture notes in Artificial Intelligence*, 2001.

[12] F. M. T. Brazier, B. M. Dunin-Keplicz, N. R. Jennings, and J. Treur, "DESIRE: Modelling multi-agent systems in a compositional formal framework," *International Journal of Cooperative Information Systems*, vol. 6, no. 1, pp. 69 – 94, 1997.

[13] M. Wooldridge, N. R. Jennings, and D. Kinny, "The Gaia methodology for agent-oriented analysis and design," *Journal of Autonomous Agents and Multi-Agent System*, vol. 3, no. 3, pp. 285 – 312, 2000.

[14] J. Lind, "Iterative software engineering for multiagent systems the massive method," *Lecture notes in Artificial Intelligenc, Springer-Verlag, Berlin*, 2001.

[15] S. Parka and V. Sugumaran, "Designing multi-agent systems: a framework and application," *Expert Systems with Applications*, vol. 28, pp. 259–271, 2005.

[16] M. Wang and C. Hung, "Extension neural network and its application," *Neural Network*, vol. 16, pp. 779 – 784, 2003.

[17] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Berlin: Springer, second ed., 1999.

[18] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford University Press, 2003.