A Multi Agent System for Monitoring Industrial Gas Turbine Start-up Sequences

Eleni E. Mangina, Stephen D. J. McArthur, J. R. McDonald, and Alan Moyes

***Abstract*—This paper introduces a novel technique for condition monitoring of gas turbine start up sequences. The vast amount of data and the complex processes behind on-line fault detection indicate the need for an automated solution. A multi agent system that views the problem as the interaction of simple independent software entities, for effective use of the available data, is presented. The overall solution is derived from the combination of partial solutions provided by the components of the multi-agent system. As a consequence, data interpretation is achieved by converting the data into appropriate information and combining individual agents' information, resulting in an automatic fault diagnosis for the engineers. This multi-agent system can employ various intelligent system techniques and has been implemented using the ZEUS Agent Building Toolkit.**

***Index Terms*—Condition monitoring, intelligent agents.**

I. Introduction

BY DEFINITION, condition monitoring is performed when it is necessary to assess the state of a plant item and to determine whether it is malfunctioning, through reasoning and observation [1]. Early diagnosis and fault identification is an important activity for maximizing the plant economy, its operational costs and levels of safety. Engineers have introduced better decision support for complex condition monitoring procedures through the application of centralized intelligent systems by using a variety of artificial intelligence (AI) techniques [2]–[5]. It is now widely recognized that problems due to the functional complexity of condition monitoring can be overcome with architectures that contain many dynamically interacting intelligent distributed modules, called intelligent agents [6]. Each agent is an autonomous system, which processes a selection of input, and the complete interpretation and diagnosis is accomplished through the process of interaction with the other agents.

This paper introduces a real time condition monitoring multi agent system (COMMAS) developed to facilitate the task of monitoring the performance of a gas turbine. The agents collect measurements from the turbine sensors and an existing engine management system (EMS), analyze abnormal measurements and perform fault diagnosis. The aim of this work is to improve the accuracy of the existing monitoring systems by building smaller software components, which contain partial information of the state of the plant being monitored. They interact in a dynamic way to support data fusion, cross sensor corroboration and decision support functions more efficiently. This contrasts with centralized applications, which model the whole plant in a single software system.

Manuscript received September 11, 2000. This work was supported by the Greek State Scholarship Foundation (I.K.Y.).

The authors are with the Center for Electrical Power Engineering, University of Strathclyde, Scotland, U.K.

Publisher Item Identifier S 0885-8950(01)06062-X.

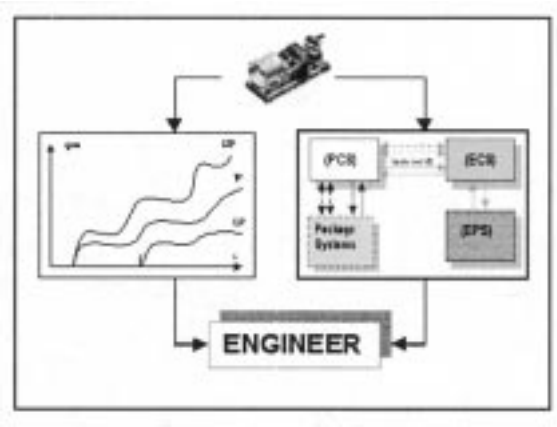

Fig. 1. Current status of problem domain.

Within this paper, Section II discusses the application problem domain of gas turbine condition monitoring. Section III presents the conceptual framework of the proposed multi-agent system and Section IV describes the analysis and design including the information flow between the agents in order to replicate the diagnostic tasks performed by the engineers. Section V presents the implementation of the framework and Section VI outlines the conclusions and some directions for future work.

II. Problem Domain

The condition monitoring of a gas turbine involves continuous measurements and evaluation of the state of the turbine during its different phases (start up, operation, maintenance etc.). Within the proposed architecture, autonomous software entities will have the ability to recognize the phase of the gas turbine operation and reason in an appropriate way based on phase specific knowledge bases (KB). Therefore, the multi-agent model will not change, but each agent will use a different knowledge base for the different phases of operation. Within this work, the software system is initially targeted for implementation during the turbine's start up phase, where knowledge has been captured and archived from the experts. The intelligent software agents focus on all the actions taken from the initial starting of a cold engine, up to the point at which synchronous speed is achieved. The Engine Management System (EMS) provides textual and graphical information to the engineers as shown in Fig. 1. There is a division of responsibility between the EMS, the Package Control System

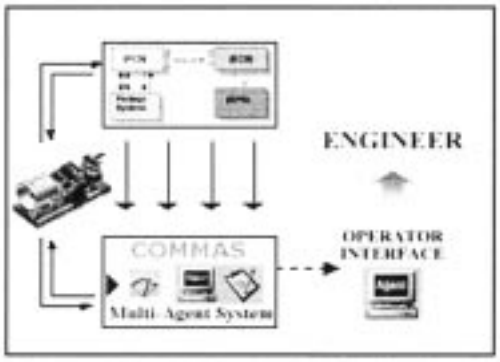

Fig. 2. Proposed solution of problem domain.

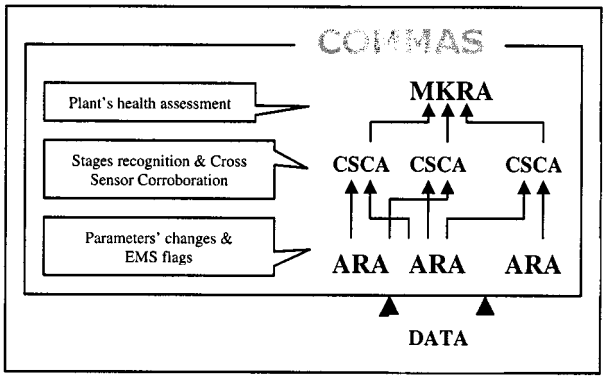

Fig. 3. Condition monitoring multi agent system.

(PCS) and the Engine Protection System (EPS). Various flags are produced from the EMS, which are either fault alarms or status indications and are passed through the EMS, PCS and EPS. These indicate the different stages of the gas turbine start up. At present, due to lack of diagnosis tools, the control system experts carry out an on-line diagnosis using the flags on display in association with the different on-line printed diagrams, which show the levels of important parameters of the gas turbine at each time point (e.g., HP, IP and LP spool speeds, which represent the speeds of the high, intermediate and low pressure within the turbine respectively). The complexity of the engineers' tasks enhances the requirement for a decentralized intelligent system, which will effectively utilize the different sources of data by employing various reasoning techniques. The centralized intelligence approach lacks flexibility and extensibility. The goals and reasoning activities tend to be fixed. Moving to an agent-based architecture allows simultaneous complex tasks to be performed in real-time; better handling of inaccurate data is achieved and each agent can be independently updated. Therefore, this emphasizes the need to use intelligent agents [7], autonomous distributed software entities, that have the ability to use different artificial intelligence techniques and to communicate and interact to solve a complex engineering problem such as condition monitoring.

In contrast to the traditional scenario of Fig. 1, the proposed decentralized, automatic solution is shown in Fig. 2. The new software is based upon intelligent agents, which efficiently supervise and control the turbine systems on behalf of the experts. The agents have to operate in a complex, dynamic and large physical system, where global solutions have to be achieved (e.g., fault diagnosis), therefore different problem solving techniques must be used to effectively analyze and process various sources of data. The software components process their data resources locally and interact to communicate their results in order to produce a coherent solution.

The inference mechanism embodied within each agent has been derived from structured knowledge elicitation meetings with gas turbine control system experts. Concise and focused information about the plant and its processes was provided in the form of transcripts and knowledge models, which have been used for the specification of the intelligent system. The connection between the multi agent system and the experts is achieved by the Operator Interface agent, which addresses the results of the software process and the expert can have a global view of the state of the plant and access the information flow within the software, during its execution. The tasks for condition monitoring have been distributed to the agents, based on the type of problem domain functions they have to perform, which are described within the following section. Importantly, the agent-based architecture allows any reasoning paradigm to be used by each individual agent. Models of the turbine are being investigated for inclusion in an agent, which uses model based reasoning. In addition, case based reasoning and artificial neural networks are also being researched.

III. Conceptual Framework

The vast amount of knowledge, the variety of condition monitoring tasks and the continuous interactions between software modules indicate the use of a multi agent system, which can access, filter and combine different types of information from various sources. The hierarchical structure of the multi agent system is the result of the functionality of the condition monitoring process. At the first stage an analysis of both the flags provided from the EMS and the raw data concerning the HP, IP and LP spool speeds has to be performed. Subsequently the results of this analysis are corroborated, as parameters must have certain relationships at certain stages of the turbine's start up. The final diagnosis is the outcome of the whole procedure, where different types of problems can be diagnosed. COMMAS hierarchical layered approach, as shown in Fig. 3, includes three different layers: Attribute Reasoning Agents (ARA), Cross Sensor Corroboration Agents (CSCA) and Meta-Knowledge Reasoning Agents (MKRA). ARA, using local problem solving techniques, will perform the interpretation of plant sensor data for the main parameters of the turbine (e.g., HP spool speed) based on their embedded knowledge (derived from the elicitation process with experts) about their constraints and thresholds during certain stages of the gas turbine start up. The next layer of agents (CSCA) will corroborate the results of the previous interpretation by asking for confirmatory evidence of problems, through communication, and they will send their conclusions to the MKRA. The final conclusion concerning the general state of the plant, details about which stage of the start up the turbine is in, any faults (e.g., sensor faults) that might have occurred during the process of condition monitoring, as well as

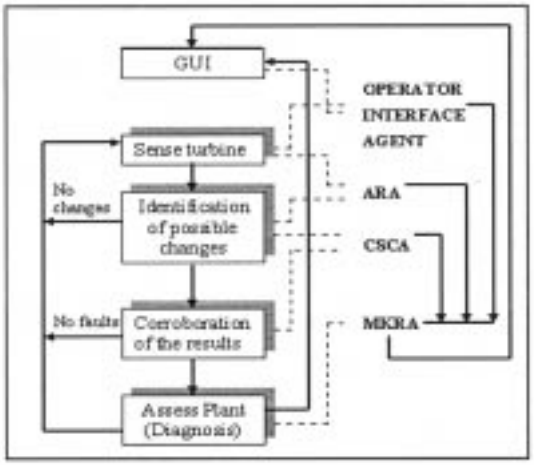

Fig. 4. General COMMAS activity diagram.

an explanation of why they occurred are given to the user by the Operator Interface agent. Currently the engineers have to derive this without any software assistance. The analysis and design of the system can be thought of as the first process of specifying the models of the system to be constructed and are given in detail in the next section.

IV. System Analysis and Design

The objective of the analysis stage is to develop and understand the software system and its structure. This understanding within the problem domain is captured in the system's organization. We view the multi agent system as a collection of roles that represent certain relationships to one another and take part in systematic interactions. Therefore, to define the organization, we must define the roles, and how these roles relate to one another and how they interact. The agents are objects that have their own beliefs (facts that represent state of the plant, updated appropriately after each sensing action), desires (goals they have to achieve) and intentions (tasks they have to execute to accomplish their goals). This belief, desire and intentions (BDI) model of agents is a formal theoretical approach, uses modal logic for handling events occurring during the plant's operation and has been applied successfully for a number of other applications [8]. The responsibilities of each agent determine the functionality and the roles associated with them. For example a role might be the ability to read a particular item of information (i.e., HP spool speed on-line data), to provide the user a graphical view of the state of the plant, etc. Two general types of agents have been developed:

- Interface agents (i.e., Operator Interface agent, to control the interaction between the user and the system)
- Application agents (i.e., ARA, CSCA, and MKRA), to control the execution of the different application tasks.

There are inevitably dependencies and relationships between the various agents; therefore interactions need to be captured in such a way that they mimic the expert's reasoning. These links between roles are represented in Fig. 4, where it is shown that the functionality determines the interaction model between the

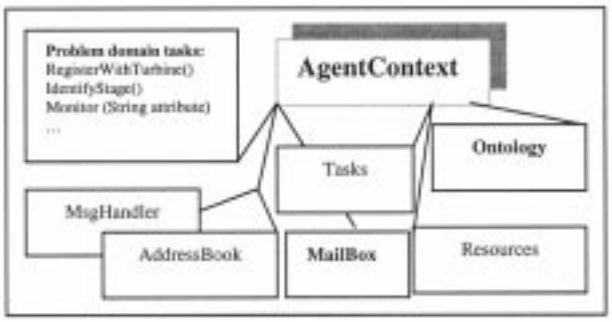

Fig. 5. Agent model.

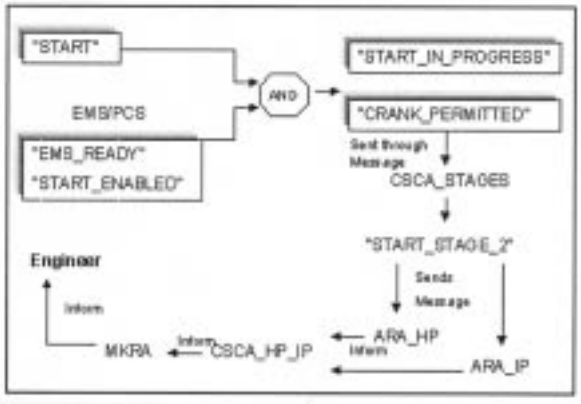

Fig. 6. Rule based interaction of agents.

agents. This pattern of interaction has been formally defined and abstracted from the sequence of experts' execution steps.

The configured agent responsibilities and interactions between them form the basis of the application design, which includes the low-level abstraction of programming code to be implemented. The agent paradigm is an extension of the object-oriented one and as a result the agents must encapsulate the required techniques to realize the application goals and communicate their results to other agents. The design process involves generating the individual agents and a general model of them is given in Fig. 5. Every agent has its own abilities, resources (facts), tasks (the problem domain tables and the general agent tasks), mailbox (to send and receive information from the other agents) and ontology [9] (all the agents within the society must refer to the same meaning for concepts they communicate i.e., a dictionary).

An example of the experts' knowledge, which has been embedded within the agents, based on which the multi-agent software system "senses" the turbine and tries to detect any changes, is given in Fig. 6. The engineer has started COMMAS (flag "START" is generated) and the EMS generates the flags "EMS_READY" and "START_ENABLED." If after a certain time period the flags "START_IN_PROGRESS" and "CRANK_PERMITTED" are given from the EMS then it is recognized from the CSCA_STAGES (which is responsible for the turbine's stage recognition) that the turbine has started stage 2 (Cranking). A message is sent then to ARA_HP and ARA_IP because they have to enter the society to monitor the HP and IP spool speeds respectively that appear at this stage.

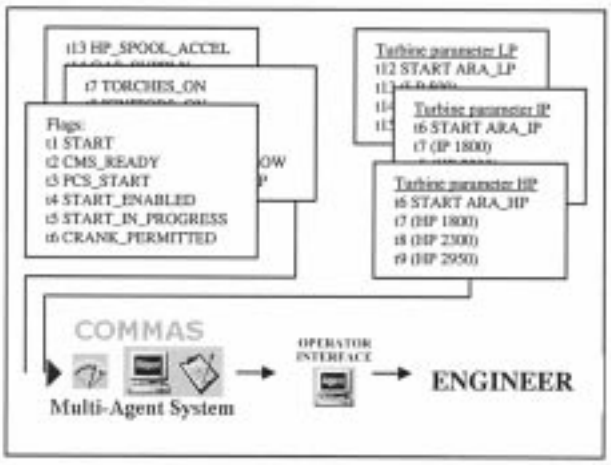

Fig. 7. Information provided to COMMAS.

TABLE I
BDI KNOWLEDGE REPRESENTATION WITHIN AGENTS

CSCA_HP_IP	MKRA
BELIEFS	***BELIEFS***
(HP_spool_speed (t7 ok))	(stage1 (started true) (finished true))
(IP_spool_speed ((t7 ok)),...	((HP_IP(status ok)), t7)
	(stage2 (started true) (finished ?)) ...
DESIRES	***DESIRES***
(HP_IP (status ?), t7)	(Turbine_state (t ?))
(HP_IP (status ?), t8) ...	(Fault_occurred (t ?)) ...
INTENTIONS	***INTENTIONS***
L((HP_spool_speed ok), t7) ^ L((IP_spool_speed ok), t7) ^ L((HP_IP ok), t7) ➜ ❑ ((HP_IP (status ok)), t7)	L(stage1(started true) (finished true) ^ L((HP_IP (status ok)), t-1) ^ ¬M((faultStage1),t-1) ➜ ❑ ((stage1 finished), t) ^ ❑ ((proceed next_stage), t)

They inform the CSCA_HP_IP of any changes, which will then send the results to the MKRA. The latter will notify the engineer of the final conclusions of the software.

The key approach to the system's analysis and design that has been followed is to go systematically from a set of requirements to a detailed design of a multi agent system for a direct implementation, which is described in the next section.

V. IMPLEMENTATION

This section illustrates how the conceptual framework has been applied based on the analysis and design of the multi-agent based system for condition monitoring of the start up sequence of a gas turbine. The start sequence can be divided into six sections (pre-start permissives, cranking, torch light up, central light up, acceleration to sub-synchronous idle and acceleration to synchronous idle) that the software must recognize (based on the different flags provided from the EMS). In parallel, it must monitor the sensor attributes. The intelligent software agents have been developed based on their specialized functionality within the software. For example the three main attributes HP, IP and LP spool speeds are being monitored from ARA_HP, ARA_IP and ARA_LP agents respectively, whereas the stage recognition is being achieved from the specialized CSCA_STAGES, which receives the flags from the EMS of the turbine.

Since the problem domain has been viewed as a set of interactions, attention has been focused at the purpose of the interactions and the message exchanges between the agents. The agents within COMMAS communicate using the Knowledge Query and Manipulation Language (KQML) [10] structure, which is of type: Send_message (**sender** MKRA)(**receiver** ARA_HP) (**type** inform) (**context** ((name ARA_HP) (register))) where the MKRA sends an inform type message to ARA_HP to order that the latter is registered within the society and starts monitoring the HP spool speed. The information COMMAS receives (flags from the EMS and data for HP, IP, and LP spool speeds) at certain time points (represented as $t_1, t_2, \ldots t_n$) is shown in Fig. 7.

Based on these facts the internal reasoning of the agents includes processing changes, events and their relations within time intervals. Logic formulae have been used to represent this procedure within the BDI framework. The modal logic operators that have been used are translated as:

L (φ, t): It is believed that "φ" occurred at time t
M (φ, t): It is possible that "φ" occurred at time t
$\Box$ (φ): It is necessary that "φ" occurs

where "φ" is any kind of fact, given or produced by the agents. As shown in Table I, both the CSCA and MKRA have certain beliefs, based on the data they receive. The ARA_HP and ARA_IP (which have processed the data sent from the HP and IP spool speed sensors of the turbine respectively) send the appropriate messages to the CSCA_HP_IP, which represent their beliefs. The latter will have to identify if the relationship between the HP and IP spool speeds is the appropriate one, so it performs its reasoning and sends the result to the MKRA, which will identify the state of the turbine process. The MKRA in conjunction with the results received from other agents (e.g., regarding which stage the turbine is in) reasons logically (based on the knowledge represented in logic formulae) to assess the state of the turbine and diagnose if a fault has occurred.

For example, in Table I, it "believes" that stage 1 has started and finished, the relationship between the HP and IP spool speeds is normal and there is no possibility ("¬" is the negation sign) of a fault existence. Therefore, MKRA informs the engineer that the first stage has finished successfully and it proceeds to the next stage, as these are necessarily true facts for it. At this stage the variables ("?" is the sign for noninstantiated variables) of the facts at its desires database will be instantiated {[Turbine_state (t normal)], [Fault_occurred (t false)]}. The beliefs, desires and intentions will also be updated. A more detailed example of the agents' interaction within COMMAS is provided in Fig. 8, where:

1) Agents need to register with the Turbine agent, which holds the database of the turbine data.
2) There are interactions between the Turbine agent and the CSCA_STAGES (to process the EMS flags) and between the Turbine agent and the ARA agents (to process the data for the turbine attributes).
3) There are interactions between the ARA_HP, ARA_IP and ARA_LP with the CSCA_HP_IP and CSCA_HP_IP_LP, after processing the information in response to the queries of the CSCA agents.

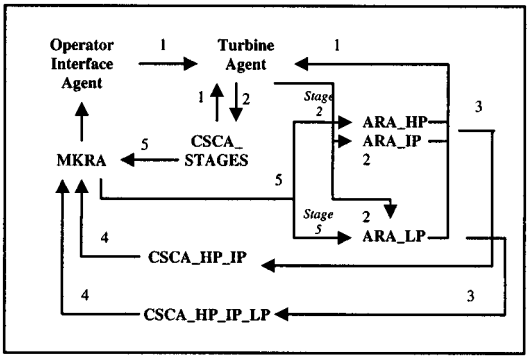

Fig. 8. Example of interactions' diagram.

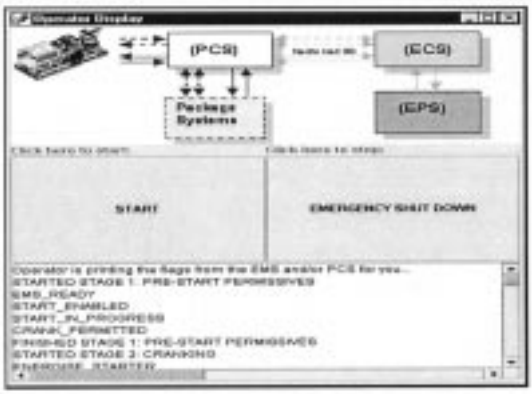

Fig. 9. Graphical user interface of operator agent.

4) CSCA agents respond to queries of the MKRA, to allow the user to view, in a diagram the attributes' changes over time.

The MKRA not only informs the user of the state of the turbine, but controls the software system (e.g., MKRA based on its belief "the first stage has finished," must inform the ARA_HP to register with the society and start monitoring the HP spool speed, as the turbine has finished the stage of pre-start permissives). Within a multi-agent environment conflicts can arise for different reasons.

Within a multi-agent environment conflicts can arise for different reasons. Agents may have conflicting beliefs or goals or they may have to share limited resources. The software agents within this work group the conflicting sets and negotiate with other agents of the society to find the exact contradiction and then, depending on their available resources, they will ask for advice from the user. Within the tasks for future development of the software system is a mechanism to deal with conflict resolution and incomplete knowledge by employing different intelligent learning techniques.

COMMAS has been implemented using the ZEUS Agent Building Toolkit [11], which is an environment for the rapid development of collaborative agent applications (within the Java Development Kit available from Sun Microsystems, Inc. [12]) and runs on all major hardware platforms. Fig. 9 illustrates the graphical user interface developed for the Operator Interface agent, where the user can visualize the flags produced from the EMS as they are sent to the MKRA for further analysis. In case

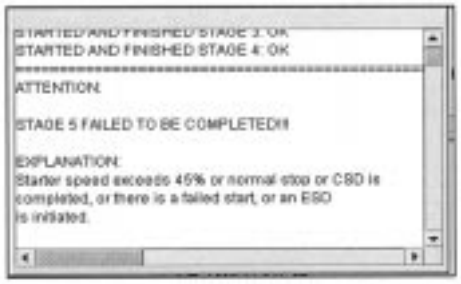

Fig. 10. MKRA explanation.

of an emergency shut down the user can also order the system to terminate.

The final results, will be given from the Operator Interface agent, an example of which is shown in Fig. 10. The start-up in this case, failed to be completed due to a failure in stage 5.

As more research is performed and more intelligence is added, the system will be able to distinguish between closely related faults. In summary, the system performs the following tasks:

- Processing of flags produced from the existing software system in order to identify the stage of operation of the turbine's start up,
- Monitoring of the HP, IP, LP spool speeds as well as other important parameters (i.e., temperature, pressure) of the turbine; and finally
- Combination of the results from these distributed processes for accurate and efficient automatic on-line condition monitoring.

VI. Discussion

In this work, the design and the implementation of a novel system has been described where the introduction of agent technology will challenge the operation of traditional condition monitoring systems. The new modular design will upgrade centralized architectures and allow on line automatic assessment of the plant's health through decentralized information processing, providing decision support for the engineers, who are currently performing the fault diagnosis manually.

The Rolls-Royce Company is currently researching agent technology with the intention of setting up test bed applications to prove the performance of real turbine systems.

Although Zeus is not a commercial quality toolkit for developing industry strength applications, it has provided promising results for an application where there were decentralized resources and a need for increased robustness and reliability. This system will be enhanced with the inclusion of more than one computational intelligence technique, which will then evaluate the results in a more efficient way. This can be implemented easily by adding new functionality to the multi agent system due to the modularity of its design.

Acknowledgment

The authors would like to thank the British Telecom Intelligent Systems Group for providing the ZEUS agent building toolkit, from which the software has been implemented.

REFERENCES

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

[2] J. R. McDonald *et al., Intelligent Knowledge Based Systems in Electrical Power Engineering*. London: Chapman & Hall, 1997.

[3] D. Gemmell, J. R. McDonald, R. W. Stewart, R. N. T. Brooke, and B. J. Weir, "A consultative expert system for fault diagnosis on turbine generator plant," *Institution of Mechanical Engineers' Proceedings Part A, Journal of Power and Energy*, Dec. 1994.

[4] S. D. J. McArthur, S. C. Bell, J. R. McDonald, R. Mather, and S. M. Burt, "Knowledge and model based decision support for power system protection engineers," in *Proceedings of the International Conference on Intelligent Systems Applications to Power Systems*, 1996, pp. 215–219.

[5] C. Booth, J. R. McDonald, and W. Hagman, "The use of artificial neural networks for the prediction and classification of vibration behavior in plant transformers," in *American Power Conference*, Chicago, 1995.

[6] C. P. Azevedo, B. Feiju, and M. Costa, "Control centres evolve with agent technology," *IEEE Computer Applications in Power*, vol. 13, no. 3, pp. 48–53, 2000.

[7] M. J. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practise," *Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995.

[8] A. S. Rao and Georgeff, "BDI agents: From theory to practice," Australian Artificial Intelligence Institute, Melbourne, Australia, 56, 1995.

[9] E. Walther, H. Eriksson, and M. A. Musen, "Plug and play: Construction of task specific expert system shells, using sharable context ontologies," Knowledge based systems laboratory, Stanford University, KSI-92-40, 1992.

[10] T. Finin, Y. Labrou, and J. Mayfield, "KQML as an agent communication language," Bradshaw Jeffrey, Software Agents, 1997.

[11] ZEUS Agent Building Toolkit (1999). [Online]. Available: http://www.labs.bt.com/projects/agents/

[12] The source for Java technology [Online]. Available: http://java.sun.com/

Eleni E. Mangina is a research student within the Center for Electrical Power Engineering at the University of Strathclyde. She received the M.Sc. degree in agricultural science from Agricultural University of Athens in 1996 and the M.Sc. degree in artificial intelligence from University of Edinburgh in 1998. Her research interests include intelligent agent applications in monitoring systems.

Stephen D. J. McArthur is the East Midlands Electricity Lecturer within the Center for Electrical Power Engineering. He received the B.Eng. (Hons) and Ph.D. degrees from the University of Strathclyde in 1992 and 1996, respectively. His research interests include Intelligent System Applications in power engineering, hybrid intelligent systems and intelligent agent technology. Dr. McArthur has 30 technical publications and is the co-editor of a book published by Chapman and Hall.

J. R. McDonald holds the Rolls-Royce Chair in Power Engineering at the University of Strathclyde. Professor James R. McDonald received the B.Sc. and Ph.D. degrees from Strathclyde University. He was appointed as the Manager of the Center for Electrical Power Engineering at Strathclyde University in July 1990 and took up the position of the Rolls-Royce Chair in Power Engineering in February 1994. His research activities lie in the areas of: power system protection and measurement; expert system applications in power engineering; energy management. He has published over 200 technical papers has published two books.

Alan Moyes holds the post of Rolls-Royce Senior Research Fellow at the Center for Electrical Power Engineering. His main research activities have been the used for knowledge engineering techniques to capture, archive, and implement expertise relating to the design, operation and monitoring of power plant and systems for on-line diagnostic and operational support systems for power engineering applications.