An Intelligent Multi Agent System for Integrated Control & Asset Management of Petroleum Production Facilities

Atalla F. Sayda[2] and James H. Taylor[1]

Department of Electrical & Computer Engineering
University of New Brunswick
PO Box: 4400, Rm. H113 Head Hall
Fredericton, NB CANADA E3B 5A3

ABSTRACT

Intelligent control and asset management for the petroleum industry is crucial for profitable oil and gas facilities operation and maintenance. A research program was initiated to study the feasibility of an intelligent asset management system for the offshore oil and gas industry in Atlantic Canada. The research program has achieved several milestones. The conceptual model of an automated asset management system, its architecture, and its behavioral model have been defined [1, 2]. Furthermore, an implementation plan for such system has been prepared, and the appropriate development tools have been chosen [3]. A system reactive agent structure was defined based on the MATLAB environment, and its communication requirements were analyzed and validated [31]. This paper builds on the previous work and proposes a general structure of the ICAM system intelligent supervisory agent and its software implementation. We also describe the software implementation using the G2 expert system development environment. Furthermore, we analyze and define the autonomy requirements of the reactive agents of such system.

1. INTRODUCTION

Asset management and control of modern process plants involve many tasks of different time-scales and complexity including data reconciliation and fusion, fault detection, isolation, and accommodation (FDIA), process model identification and optimization, and supervisory control. The automation of these complementary tasks within an information and control infrastructure will reduce maintenance expenses, improve utilization and output of manufacturing equipment, enhance safety, and improve product quality. Many research studies proposed different combinations of systems theoretic and artificial intelligence techniques to tackle the asset management problem, and delineated the requirements of such system [4], [5], [6].

Several research programs addressed the automation of asset management in large complex systems, namely the Pilots Associate (PA) program sponsored by the Defense Advanced Research Projects Agency (DARPA) [7], [8], the Rotorcraft Pilots Associate (RPA) program funded by the US army [9], MAGIC (Multi-Agent-Based Diagnostic Data Acquisition and Management in Complex Systems) developed by a joint venture of several European universities and companies [10], ISHM (Integrated System Health Management) project developed by NASA for space applications [11], AEGIS (Abnormal Event Guidance and Information System) developed by the Honeywell led Abnormal Situation Management (ASM) Consortium in the United States [12], and CHEM-DSS (Advanced Decision Support System for Chemical/Petrochemical Manufacturing Processes) developed by the European Community (EC) Intelligent Manufacturing Systems (IMS) consortium [13]. The most important project among these projects is AEGIS, which proposes a comprehensive asset management framework from an industrial view point. AEGIS built on the experience of military aviation research projects, especially the Pilots Associate (PA) and the Rotorcraft Pilots Associate (RPA) [14]. Although the 12 year old research program has achieved several goals and developed a well established abnormal situation management awareness and culture, it did not address the automation of massive process data interpretation and process fault diagnosis and accommodation, which would be aimed to minimize the workload on process operators [15].

---

1. Email: jtaylor@unb.ca, Tel: +1.506.453.5101 ; FAX: +1.506.453.3589
2. Email: atalla.sayda@unb.ca, Tel: +1.506.449.0644 ; FAX: +1.506.453.3589

In order to build on the AEGIS project experiences and to incorporate the state of the art of fault diagnosis, artificial intelligence (AI) and wireless sensor networks techniques, a new asset management research project, PAWS (Petroleum Applications of Wireless Systems) was initiated by a joint venture of Atlantic Canadian universities and the National Research Council of Canada (NRC) for oil and gas applications [1], [16], [2], [17], [18], [3], [19], [20], [21], [22], [31]. The PAWS project scope is to develop a control and information management system which consists of two subsystems. The first subsystem is a wireless sensor network which will alleviate the need for data cables in offshore oil rigs and improve flexibility for adding and reconfiguring sensors. The second subsystem intelligently manages the massive data flow from oil rigs and interprets it so as to help operators take more appropriate decisions during abnormal events and, through intelligent control, improve process economics.

As part of the PAWS project, our team is developing an *intelligent control and asset management system* (ICAM system). We defined the conceptual model of the ICAM system, its architecture, and its behavioral model [1], [2]. Furthermore, we prepared an implementation plan for such system, and chose the appropriate development tools [3]. The general structure of ICAM system reactive agents was defined, and the communication requirements were analyzed and validated [31]. This paper builds on the previous work and proposes a general structure of the ICAM system intelligent supervisory agent. The paper is organized as follows: First, we describe the structure of an ICAM system reactive agent to achieve the best autonomy in section 2. Then, we analyze the ICAM system artificial intelligence (AI) requirements, and suggest a general structure for the ICAM system intelligent supervisory agent in sections 3 and 4 respectively. Complex ICAM system knowledge representation is described in section 5. Finally, we discuss the requirements of the ICAM system knowledge processing in section 6.

2. AUTONOMY REQUIREMENTS OF ICAM SYSTEM REACTIVE AGENTS

Having proposed the ICAM system development plan in previous work [3], it is crucial to design the reactive agent structure to achieve specific autonomy requirements in terms of an overlapping scheme for communication and computation along with ease of prototyping and deployment. The Message Passing Interface (MPI) communication model meets the autonomy requirements by offering many advantages such as expressivity, ease of debugging, and most importantly high performance [23], [24]. MPI is a specification and a library which provides the infrastructure for communications among several parallel computational processes. MPI gives system designers the freedom to implement their own protocols that best fit their systems' requirements. In order to reconcile efficient computation with ease of prototyping, the ICAM system is deployed as a distributed interconnection of MATLAB computational (i.e., reactive) agents, which runs on a network of several Windows XP workstations. Distributed MATLAB sessions exchange messages by using our newly developed MPI communication protocol. Exchanged messages have two roles; a control role, to achieve internal coordination with other agents, and a numerical data processing role, to achieve the best interaction with the external environment (e.g., offshore oil processing rigs).

Figure 1 shows the structure of a reactive agent of the ICAM system. The reactive agent is implemented as a MATLAB m-script, which runs two communication tasks and a computational one. The computational task represents the agent's main functionality (e.g., model ID, fault detection and isolation, etc...). The first communication task is an MPI remote memory access (RMA) protocol, which provides the basic buffered messaging capabilities with minimum overhead. Furthermore, a public memory window is embedded in the protocol for remote access by other agents. The memory window will act as a black board for direct transfer of complex numerical data structures among agents. This design decision was made after investigating the advanced features of the newest MPI 2.0 library [25], and to meet the blackboard functionality described in the behavioral model of the ICAM system. The second communication task manages the connection with the main system supervisor (implemented as G2 expert system). The proposed agent structure paves the way to design and to rapid prototype any complex multi-agent system for many applications. This definitely enables system designers to implement any communication protocol in addition to exploiting the full power of the MATLAB simulation, computation and development environment.

3. ARTIFICIAL INTELLIGENCE REQUIREMENTS OF THE ICAM SYSTEM

The artificial intelligence (AI) requirements of the ICAM system have to address different issues such as coordinating the system's internal behavior (i.e., how the different agents interact) and managing the external manufacturing process. The choice of the appropriate AI paradigm is very crucial to high performance and real-time issues. Different AI paradigms (e.g., rule based expert systems, case based reasoning (CBR) systems, neural nets (NN)...) have different strengths and disadvantages. Another issue is the intelligence distribution; locally within each

agent versus globally within the system. Obviously knowledge specific to agent activity will be embedded at the agent level. However, global system intelligence should address the internal system coherence and its external interaction with the environment. An intelligent ICAM supervisory agent is responsible of managing the global system internal and external behavior.

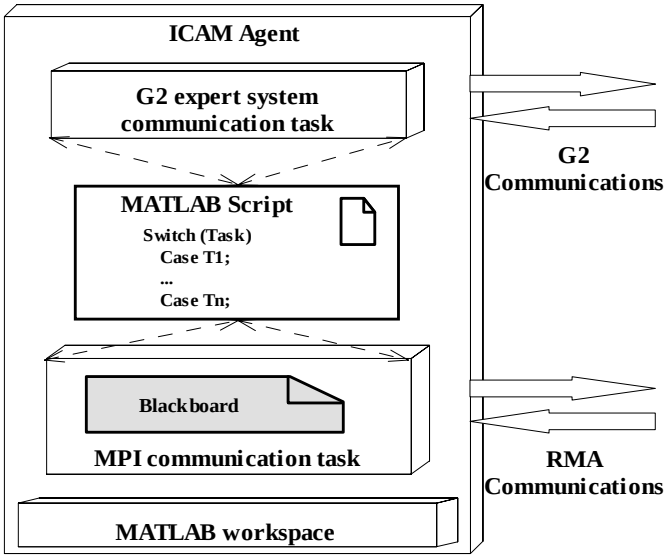

Fig. 1. ICAM system reactive agent deployment structure

When it comes to selecting an appropriate AI paradigm, we were at first attracted by the case-based reasoning (CBR) approach, as it promised to meet the high performance, learning and real-time requirements of the ICAM system [1], [2]. The CBR paradigm is a novel problem-solving strategy and machine learning technique. In principle, it solves problems by retrieving a "nearest neighbor" past problem from its case base, evaluating any differences, and adapting the past problem solution to handle the new circumstances. Every new problem that is handled successfully is added to the case base; if the new solution is a failure that information is also stored. While this approach is apparently systematic and easily automated, there are several major drawbacks: (1) developing an algorithm to extract a "nearest neighbor" problem is domain-specific and may be very difficult, and (2) "adapting the past problem solution to handle the new circumstances" is also much easier said than done. Although many CBR programs were developed during the 80's and mid 90's [27], the CBR development process slowed greatly due to the problems mentioned above (and others), so we are shifting to another paradigm.

Traditional rule-based systems have a few well-known drawbacks, such as difficult knowledge acquisition, lack of a memory of tackled problems or previous experience, poor inference efficiency, ineffectiveness in dealing with exceptions and novel situations and lack of learning mechanisms, to name a few. However, the development of new software standards and technologies for rule-based expert systems continued to progress. Such development has enabled rule-based expert systems to overcome many of their drawbacks, and to compete with the CBR AI paradigm. In fact, if we can limit ourselves to "crisp" problems then the "nearest neighbor" problem does not arise, and we can use rules to define a case base, and retrieve and implement solutions. Semi-automated procedures can also be implemented to allow operators or process engineers to enter new cases and thus implement a limited form of learning.

Among the commercially-available rule-based expert system shells, the G2 real-time expert system shell from Gensym Corporation is considered the most versatile real-time expert system shell, as it integrates many software technologies and standards [28]. The G2 platform uniquely combines real-time reasoning technologies, including rules, work flows, procedures, object-oriented modeling, simulation, and graphics, in a single development and deployment environment. G2 can transform real-time operations data into automated decisions and actions, and can maintain an understanding of the behavior of processes over time.

Several attempts were made to integrate the G2 expert system shell with computational modules for fault diagnosis. Although the AEGIS and CHEM-DSS research projects partnered with the Gensym Corporation, their research publication did not indicate how G2 servers are implemented in their systems. The ISHM research project supported by NASA used six G2 servers to monitor International Space Station (ISS) subsystems, including the

mechanical, structural, electrical, environmental and computational systems. The G2 servers continually inspect and analyze data transmitted from space during missions. Cinar *et al* have successfully combined multivariate statistical data analysis with expert systems for process fault diagnosis. Basically the multivariate statistical data analysis module developed in MATLAB was converted into C code, and then linked with G2 expert system through a G2 standard interface (GSI) link [32], [34]. Cinar *et al* exploited only the G2 diagnostic assistant (GDA) capability (i.e., a graphical design tool similar to Simulink/MATLAB). Thornhill *et al* used the Computer Aided Engineering Exchange (CAEX) IEC/PAS 62424 standard for representation of process information in XML. The CAEX Plant Analyzer prototype incorporates process knowledge by linking the plant topology and a simple reasoning engine developed in Prolog with the results from plant disturbance signal-based analysis [33].

The integration of the G2 expert system development environment with the ICAM system would benefit from and build on the previous G2 integration attempts. This would enhance the whole ICAM system performance, and enable the ICAM system to intelligently coordinate its internal behavior and interact with the external industrial process as well. The G2 development environment offers a goal-based rapid prototyping design, in which requirements analysis, design, and development tasks are done simultaneously and incrementally during the ICAM system development life cycle. G2 also adheres to software development standards such as object-oriented design, modularity, reusability, scalability, application programmer's interface (API), and user interface standards [28]. To meet such software requirements during the design and development of the ICAM system supervisory agent, AI design requirements such as the supervisory agent structure and knowledge representation and processing have to be determined.

4. ICAM SYSTEM SUPERVISORY AGENT IMPLEMENTATION

Modules are the building blocks of complex G2 applications. A modular knowledge base (KB) consists of multiple G2 modules. The modules that make up an application form a module hierarchy, which specifies the hierarchical dependencies between modules [28]. Decomposing a large project into multiple small modules allows developers to divide and merge work. Modules can be structural or functional ones. The structural modules contain classes or capabilities that need to be shared in large applications; functional modules implement well defined goals. The ICAM system supervisory agent, which is a very complex artificial intelligence application, forms a good candidate for the modularization design approach. While the modularization design approach may add some overhead on the overall performance of the agent, it effectively organizes knowledge, and simplifies the development and deployment processes.

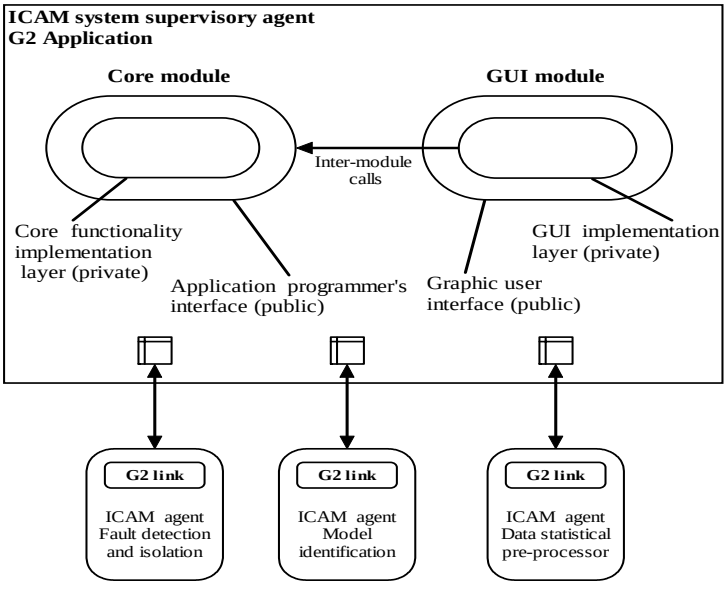

Fig. 2. ICAM supervisory agent architecture

To meet the module reusability requirement, the guidelines for G2 application development recommend use of a four layer, two-module architecture, in which the graphic user interface (GUI) is in a separate module. Figure 2 illustrates the general architecture of the ICAM supervisory agent, which accordingly has two modules. The first

module contains the agent's core functionality implementation layer and its application programmer's interface (API) layer, which protects the internal data structures in the core from corruption by other modules. The second module contains the public graphical user interface (GUI) layer and its GUI implementation layer, which interacts directly with the first module through its API layer. The ICAM system supervisory agent interacts with the other reactive agents through their external G2 links. The internal states of the ICAM system agents and the external environment are communicated to enable the supervisory agent to reason and make the correct and appropriate decisions for better system management.

5. KNOWLEDGE REPRESENTATION IN THE CORE LAYER:

ICAM system supervisory agent contains multi-faceted complex knowledge such as the internal structure of the ICAM system and the structure of the external environment (e.g., manufacturing plant topology, enterprise business structure). To represent such complex knowledge, organizing the knowledge structure in the core layer of the supervisory agent as a hierarchy of smaller modules would be the solution, as shown in figure 3. Each module is represented in the G2 development environment as a knowledge base (KB). The modular knowledge base design approach supports objected-oriented design principles, increases productivity, encourages code reuse and scalability, and improves maintainability.

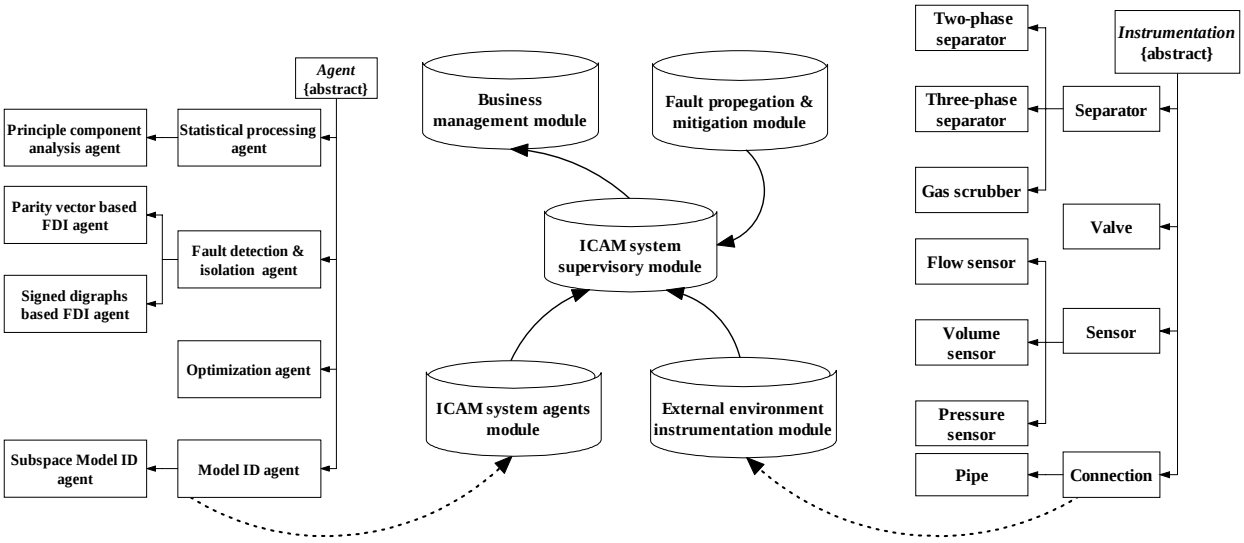

Fig. 3. Knowledge representation structure in the ICAM supervisory agent

The core (private layer) of the ICAM system supervisory agent has five modular knowledge bases (kb), which are organized as a module hierarchy. Basic knowledge about the ICAM system elements is represented in three lower level knowledge bases (i.e., ICAM system agents' kb, external environment instrumentation kb, and fault propagation and mitigation kb). The first knowledge base organizes the different conceptual agents of the ICAM system (e.g. fault detection and isolation agents, optimization agent, etc ...). In contrast, the second knowledge base maps the external environment physical instrumentation (e.g., valves, sensors, and other chemical process equipment) into its class hierarchy. Instrumentation and process faults and their mitigating actions are represented as classes in the third knowledge base. Each basic element (i.e., object) in these knowledge bases has properties to represent its physical or conceptual characteristics; and has methods to represent its behavior. Elements are further organized as a class hierarchy to exploit object-oriented standards such as abstraction, inheritance, and information hiding and encapsulation. An abstract class, which hides its basic properties and methods, is first designed. More properties and methods are added to the higher level classes in the class hierarchy, which inherit from the abstract class.

The ICAM system supervisory knowledge base merges the knowledge from the lower level modules into a three-layer knowledge base, where each layer represents a subsystem of connected objects (i.e., classes), as illustrated in figure 4. The first layer (i.e., the ICAM system structure layer) assembles the conceptual structure of the ICAM system from the agent class hierarchy of the lower level knowledge base. This layer is responsible of managing the internal behavior of the ICAM system. Fault propagation and mitigating actions are assembled into object trees, and

mapped into the second layer, which manages the external environment during abnormal situations. In fact, it isolates instrumentation faults, and presents their propagation maps and their appropriate migrating actions to process operators. The third layer (i.e., process topology layer) represents the external process topology, where different process instrumentation objects are used from the instrumentation knowledge base module. Other knowledge bases can be added to represent other types of knowledge such as the enterprise business management module.

6. ICAM SYSTEM KNOWLEDGE PROCESSING:

The G2 development environment offers several programming constructs for processing data such as procedures, methods, and rules. Procedures, which are independent of any class, define a general functionality, and can be invoked by rules, other procedures, and GUI actions. In contrast, methods whose invocation and structure are similar to procedures, define classes' behavior. Methods structure the behavior of different objects through method inheritance between different classes of the class hierarchy. Procedures and method invocation can be single-threaded (i.e., sequential) or multi-threaded (parallel). Rules, which are statements with antecedent and consequent parts, represent expert decision-making power by reasoning over data or event conditions ("facts"). Rules are invoked by forward chaining, backward chaining, or scanning.

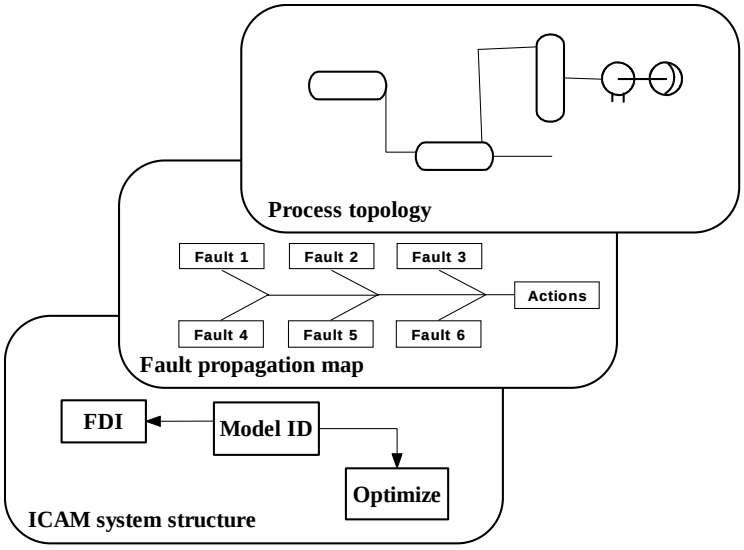

Fig. 4. Layers of the ICAM system knowledge base module

Forward chaining is a form of deductive reasoning, in which rules are invoked to attempt to draw conclusions from existing facts. If the antecedent of a rule is true, then its consequent part is executed to create new facts, thus causing a chain of rule invocations in a rule base. Backward chaining is the process of determining a value of a goal or a variable by looking for rules that conclude that goal. Backward chaining follows two different search strategies. The breadth first search strategy examines all rules that could determine the value of the current goal and sets their antecedents as sub-goals before backtracking through other rules to determine the validity of each sub-goal. In contrast, the depth first search strategy backtracks through all of the rules in a knowledge base that could lead to determining the value of a goal by a single rule. Scanning is another rule invocation method in which rules are invoked automatically based on a fixed, user-defined frequency.

A G2 rule-based system maps out a multi-threaded path of execution, which is potentially different each time the rule is invoked. For this reason, rule-based systems are often more complex, harder to test, debug, and maintain, and less efficient than procedure-based systems based on methods. Thus, rules should be used for specific purposes such as general event detection and event detection based on data driven processing and forward chaining [28]. Since the ICAM system knowledge is multi-faceted and complex, its knowledge processing structure should be also distributed and organized according to the class and/or module hierarchy of the supervisory agent. For example, generic rules for event detection of a specific reactive agent can be organized in the class associated with that reactive agent. Rules can also be categorized to achieve certain functionality. For example, the fault propagation and

mitigation schemes (i.e., cases) can be implemented as a rule category. This would narrow the scope of rules, where rules are only applied to their specified level in the class hierarchy and/or the module hierarchy. Consequently, rules invocation by forward chaining will be less prone to errors. The distribution of knowledge representation and processing would meet most of the software requirements. This would pave the way for managing complex process plants by dividing it into sub-processes that can be managed by a separate ICAM system. A universal supervisor can then manage the whole hierarchy of sub-processes efficiently.

7. CONCLUSIONS

As part of the PAWS project, we have demonstrated good progress in the design and development of the ICAM system. The ICAM reactive agents' structure was designed to achieve autonomy requirements in terms of overlapping scheme efficient for communication and computation. In order to guarantee a robust and coherent system performance, the ICAM system's artificial intelligence requirements, structure and tools were suggested. The software implementation of the ICAM intelligent supervisory agent was discussed. Complex ICAM system knowledge representation and processing requirements were thoroughly analyzed. The implementation and validation of the AI requirements in an ICAM system prototype will be the cornerstone of future work. We believe that the ICAM system will pave the way to real intelligent multi-agent systems for many applications.

ACKNOWLEDGEMENT

This project is supported by the Atlantic Canada Opportunities Agency (ACOA) under the Atlantic Innovation Fund (AIF) program. The authors gratefully acknowledge that support and the collaboration of the Cape Breton University (CBU), the National Research Council (NRC) of Canada, and the College of the North Atlantic (CNA). The authors also acknowledge the support of Natural Sciences and Engineering Research Council of Canada (NSERC) for funding the first author's research.

REFERENCES

[1] J. H. Taylor and A. F. Sayda, "Intelligent information, monitoring, and control technology for industrial process applications," in The 15th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM), Bilbao, Spain, July 2005.

[2] J. H. Taylor and A. F. Sayda, "An intelligent architecture for integrated control and asset management for industrial processes," in Proc. IEEE International Symposium on Intelligent Control (ISIC05), Limassol, Cyprus, June 2005.

[3] A. F. Sayda and J. H. Taylor, "An implementation plan for integrated control and asset management of petroleum production facilities," in IEEE International Symposium on Intelligent Control ISIC06. Munich, Germany: IEEE, October 4-6 2006.

[4] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, "A review of process fault detection and diagnosis part 1, 2, 3," Computer & Chemical Engineering, vol. 27, no. 3, pp. 293–346, 2003.

[5] R. J. Patton, "Fault-tolerant control systems: The 1997 situation," in IFAC Symposium on Fault Detection Supervision and Safety for Technical Processes, R. J. Patton and J. Chen, Eds., vol. 3. Kingston Upon Hull, UK: IFAC, August 1997, pp. 1033–1054.

[6] P. M. Frank and B. Köppen-Seliger, "New developments using AI in fault diagnosis," Engineering Applications of Artificial Intelligence, vol. 10, no. 1, pp. 3–14, 1997.

[7] R. L. Small and C. W. Howard, "A real-time approach to information management in a pilot's associate," in Proceedings of Digital Avionics Systems Conference. IEEE/AIAA, 14–17 Oct 1991, pp. 440–445.

[8] S. B. Banks and C. S. Lizza, "Pilot's associate: a cooperative, knowledge-based system application," IEEE Expert, vol. 6, no. 3, pp. 18–29, June 1991.

[9] C. A. Miller and M. D. Hannen, "Rotorcraft pilot's associate: Design and evaluation of an intelligent user interface for cockpit information management," Knowledge-Based Systems, vol. 12, no. 8, pp. 443–456, Dec 1999.

[10] B. Köppen-Seliger, T. Marcu, M. Capobianco, S. Gentil, M. Albert, and S. Latzel, "MAGIC: An integrated approach for diagnostic data management and operator support," in Proceedings of the 5th IFAC Symposium Fault Detection, Supervision and Safety of Technical Processes - SAFEPROCESS05, Washington D.C., 2003.

[11] J. Schmalzel, F. Figueroa, J. Morris, S. Mandayam, and R. Polikar, "An architecture for intelligent systems based on smart sensors," IEEE Transactions on Instrumentation and Measurement, vol. 54,no. 4, pp. 1612–1616, August 2005.

[12] T. Cochran, P. Bullemer, and I. Nimmo, "Managing abnormal situations in the process industries parts 1, 2, 3," in NIST Proceedings of the Motor Vehicle Manufacturing Technology (MVMT) Workshop, Ann Arbor, MI, 1997.

[13] S. Cauvin, "Chem: Advanced decision support system for chemical/ petrochemical manufacturing processes," in CHEM Project Annual Meeting. Lille, France: http://www.chem-dss.org/, 25-26 March 2004.

[14] E. L. Cochran, C. Miller, and P. Bullemer, "Abnormal situation management in petrochemical plants: can a pilot's associate crack crude," in Proceedings of the 1996 IEEE National Aerospace and Electronics Conference, NAECON, vol. v2. Dayton, OH, USA: IEEE, Piscataway, NJ, USA, May 20-23 1996, pp. 806–813.

[15] A. Ogden-Swift, "Reducing the costs of abnormal situations: the next profit opportunity," in IEEE Advanced Process Control Applications for Industry Workshop (APC2005), Vancouver, Canada, May 2005.

[16] M. Omana and J. H. Taylor, "Robust fault detection and isolation using a parity equation implementation of directional residuals," in IEEE Advanced Process Control Applications for Industry Workshop (APC2005), Vancouver, Canada, May 2005.

[17] W. Larimore, in Multivariable System Identification Workshop. Fredericton, New Brunswick: University of New Brunswick, 31 October – 2 November 2005.

[18] C. Smith, C. Gauthier, and J. H. Taylor, Petroleum Applications of Wireless Sensors (PAWS) Workshop. Sydney, Nova Scotia: Cape Breton University, 22–23 August 2005.

[19] A. F. Sayda and J. H. Taylor, "Modeling and control of three-phase gravity separators in oil production facilities," accepted by the American Control Conference (ACC), New York, NY, 11-13 July 2007.

[20] M. Omana and J. H. Taylor, "Enhanced sensor/actuator resolution and robustness analysis for FDI using the extended generalized parity vector technique.", in Proc. American Control Conference (ACC). Minneapolis, Minn. AACC, 14-16 June 2006.

[21] J. H. Taylor and M. Laylabadi, "A novel adaptive nonlinear dynamic data reconciliation and gross error detection method," in IEEE Conference on Control Applications, Munich, Germany, October 4-6 2006.

[22] M. Laylabadi and J. H. Taylor, "Anddr with novel gross error detection and smart tracking system," in 12th IFAC Symposium on Information Control Problems in Manufacturing, Saint-Etienne, France, May 17-19 2006.

[23] W. Gropp, E. Lusk, and A. Skjellum, Using MPI: portable parallel programming with the message-passing interface, 2nd ed., ser. Scientific and Engineering Computation, J. Kowalik, Ed. Cambridge, Massachusetts: MIT Press, 1999.

[24] W. Gropp and et al, MPI: The Complete Reference. The MIT Press, 1998, vol. 2.

[25] W. Gropp, E. Lusk, and R. Thakur, Using MPI-2: Advanced features of the message-passing interface, ser. Scientific and Engineering Computation. Cambridge, Massachusetts: MIT Press, 1999.

[26] W. Gropp and E. Lusk, "Tuning MPI applications for peak performance," www.mcs.anl.gov/Projects/mpi/tutorials/perf, Argonne National Laborary.

[27] K.-D. Althoff, E. Auriol, R. Barletta, and M. Manago, "A review of industrial case-based reasoning tools," Al intelligence, Oxford, UK, Tech. Rep., 1995.

[28] G2 for Application Developers Reference Manual, 8th ed., Gensym Corporation, Burlington, Massachusetts, December 2005.

[29] N. Viswanadham, J. H. Taylor, and E. C. Luce, "A frequency domain approach to failure detection and isolation with application to GE21 turbine engine control system," Control Theory and Advanced Technology, vol. 3, no. 1, pp. 45–72, 1987.

[30] M. Omana. "Robust fault detection and isolation using a parity equation implementation of directional residuals," Master's thesis, University of New Brunswick, 2005.

[31] A. F. Sayda and J. H. Taylor, "Toward a practical multi-agent system for integrated control and asset management of petroleum production facilities," submitted to the IEEE International Symposium on Intelligent Control, Singapore, 1-3 October 2007.

[32] A. Norvilas, A. Negiz, J. DeCicco, and A. Cinar, "Intelligent process monitoring by interfacing knowledge-based systems and multivariate statistical monitoring.", Journal of Process Control, 10, pp. 341–350, 2000.

[33] S.Y. Yim, H.G. Ananthakumar, L. Benabbas, A. Horch, R. Drath, N.F. Thornhill, "Using process topology in plant-wide control loop performance assessment.", Computers and Chemical Engineering, 31, pp. 86–99, 2006.

[34] E. Tatara, and A. Cinar, " An intelligent system for multivariate statistical process monitoring and diagnosis.", ISA Transactions, 41, pp. 255–270, 2002.