Toward A Practical Multi-agent System for Integrated Control and Asset Management of Petroleum Production Facilities

Atalla F. Sayda and James H. Taylor

Abstract— **This paper addresses a practical intelligent multi-agent system for asset management for the petroleum industry, which is crucial for profitable oil and gas facilities operations and maintenance. A research project was initiated to study the feasibility of an intelligent asset management system. Having proposed a conceptual model, architecture, and implementation plan for such a system in previous work [1], [2], [3], we define the autonomy, communications, and artificial intelligence (AI) requirements of the component agents of such a system. We also discuss the software implementation of such agents. Furthermore, we describe a simple system prototype, and conduct a real time simulation experiment to analyze the prototype performance. Simulation results reveal that** MATLAB **can be used to build high performance real-time multi-agent systems, which can be used for many applications.**

I. INTRODUCTION

Asset management and control of modern process plants involves 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-CSS (Advanced Decision Support System for Chemical/Petrochemical Manufacturing Processes) developed by the European Community (EC) Intelligent Manufacturing Systems (IMS) consortium [13].

Among all projects, AEGIS is the most important one, 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].

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], [23], [24]. 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. Wireless sensors will be used where permitted by safety. 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) in which several milestones have been achieved. 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]. This paper builds on the previous work and proposes a general ICAM system agent structure, and further defines the communication and the artificial intelligence requirements of the ICAM system. Furthermore, a real time simulation experiment is conducted on a prototype of the ICAM system to test and confirm its performance.

The paper is organized as follows: First, we describe the structure of an ICAM system agent to achieve the best autonomy in section 2. Then, we analyze the ICAM system

James H. Taylor is with the Department of Electrical & Computer Engineering, University of New Brunswick, PO Box 4400, Fredericton, NB CANADA E3B 5A3 jtaylor@unb.ca

Atalla F. Sayda is a PhD candidate with the Department of Electrical & Computer Engineering, University of New Brunswick, PO Box 4400, Fredericton, NB CANADA E3B 5A3 atalla.sayda@unb.ca

communication and artificial intelligence requirements in sections 3 and 4 respectively. Then we discuss the ICAM system agent implementation in section 5. Finally, we describe an ICAM system prototype, and discuss a real time simulation experiment of such a prototype in section 6.

II. Autonomy requirements of ICAM system agents

Having proposed the ICAM system development plan in previous work [3], it is crucial to design the 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 [25], [26]. 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 requirements, the ICAM system is deployed as distributed interconnection of MATLAB computational agents, which runs on a network of several Windows XP workstations. Distributed MATLAB sessions exchange messages by using a 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).

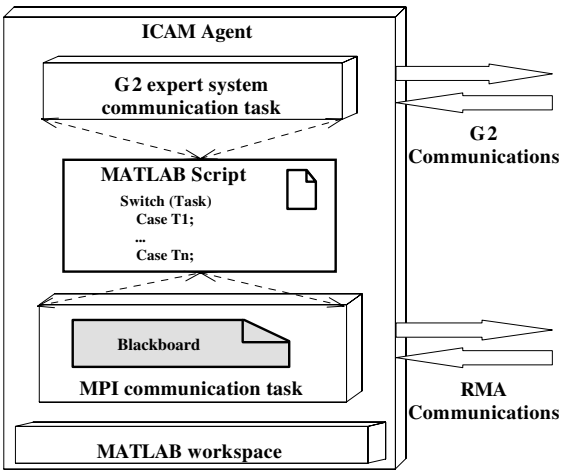

Fig. 1. ICAM system agent deployment structure

Figure 1 shows the structure of a general agent of the ICAM system. The general 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 (refer to the next section for more details). 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 [27], 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 (i.e., 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 and development environment.

III. ICAM system communications requirements

MPI communication library offers many communication modes and protocols giving system designers the freedom and flexibility to implement their communication specifications and protocols. The MPI library specifies synchronous, buffered, ready, and nonblocking communication modes. In the synchronous mode, communicating processes are blocked till a message transfer operation is completed. However, the non-blocking mode does not block the communicating processes, which allows more flexible implementation in terms of communication/computation overlap. Buffered mode gives designers more manageability over communication buffers; whereas the ready mode guarantees correct message sending operation if a matching receiving operation is posted.

Among pre-specified MPI protocols, designers can choose from several protocols such as the Eager, the Rendezvous, and the One-sided protocols for implementation. The Eager protocol can be used to send messages assuming that the destination can store. This protocol has minimal startup overhead and is used to implement low latency message passing for smaller messages. The Eager protocol has advantages in terms of programming simplicity, and reduction of synchronization delays. However, it requires significant buffering, additional buffer copies, and more CPU involvement at the destination. The Rendezvous protocol negotiates the buffer availability at the receiver side before the message is actually transferred. This protocol is used for transferring large messages when the sender is not sure whether the receiver actually has the buffer space to hold the entire message. This protocol is safe and robust, and may save in memory. However, it requires more complex programming and may introduce synchronization delays.

The One-sided protocol (i.e., remote memory access (RMA) protocol) moves data from one process to another with a single routine that specifies both where the data is coming from and where it is going. Communicating agents using this protocol must have a designated public memory (i.e., blackboard), which can be remotely accessed. This

protocol has nearly the best performance compared to others in terms of synchronization delays, however it requires a very careful synchronization planning process [27]. Having described the communication design options available in the MPI library and according to the high performance MPI recommendations [28], it is our opinion that the ICAM system communications should meet the following requirements:

- In order to avoid dead locks, synchronization time, and serialization problems, the non blocking communication mode can be used.
- To address the message size and scalability issues, the Rendezvous protocol would be the perfect candidate among the other MPI protocols.
- The problem of buffer contention and achieving fairness in message passing can be resolved by having large communication buffers.
- The one-sided protocol can be also implemented to augment ICAM system communication performance by enabling agents to have their own private blackboards, as was discussed in the previous section.

IV. Artificial intelligence (AI) requirements of 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) versus managing the external manufacturing process. The choice of the appropriate AI paradigm is very crucial to the 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 the 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 [23].

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 neighbour" 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 neighbour" 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 [29], 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 neighbour" 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 industrial 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 [30]. 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. 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. We have refined our AI requirements analysis, and designed the general structure of the ICAM system intelligent supervisory agent and its software implementation using the G2 expert system development environment [23].

V. ICAM agent implementation

Having analyzed the autonomy and communication requirement of ICAM agents, the general implementation of these agents can then be designed, as shown in figure 2.

The general ICAM agent implementation starts its main Matlab script and its associated graphic user interface (GUI). After the ICAM agent is instantiated and its buffers are initialized, the MPI communication environment and the G2 expert system link are initialized. The agent's specified computational task is started.

Once the computations are done, the communication tasks are executed based on the agent's internal state and decisions. If the agent decides that it requires further deliberation about its internal state or its response to the external environment, then messages are exchanged with the ICAM system supervisor (i.e., the G2 expert system). On the other hand, if the agent requires more data for better awareness of the external environment, then it would exchange messages with other agents through its MPI link. if the computational task is done, the task is ended; if not, the computation loop continues to

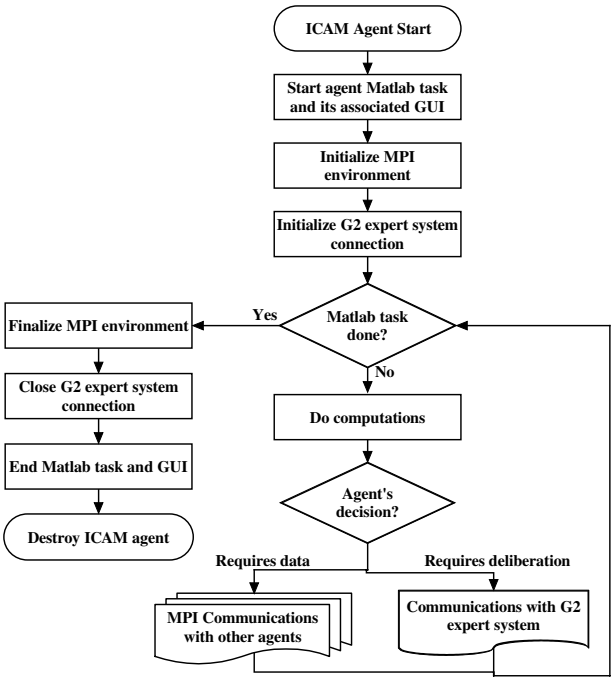

Fig. 2. ICAM agent implementation flow chart

execute. The MPI environment is finalized, and the G2 expert system link is disconnected when the ICAM system shuts down.

VI. The ICAM system prototype

In order to have the ICAM system requirements deployed in a real-world system, a prototype has to be developed. Figure 3 illustrates the simplified ICAM system prototype. Data from the external plant are received by the statistical data monitoring agent, which preprocesses the data by removing undesired discrepancies. When the data statistical preprocessor detects a change in the operating point or an abnormal change in data, it alerts the model ID and FDI agents to further identify the nature of the data change. If the change is in the process operating point, the FDI agent asks the model ID agent to update the process model parameters. If the change is a process fault (i.e., a sensor or actuator fault), the FDI agent detects the nature of the fault and notifies the ICAM system supervisor for further processing. For every event that occurs, the supervisor is notified, which in turn monitors and assesses the logical behavior of the system.

The FDI agent exploits the generalized parity space (GPS) to generate a set of directional residuals, from which process faults can be determined [31], [16], [20], [32], [24]. The model ID agent implements the recursive least squares (RLS) with forgetting factor technique. The supervisory agent is a G2 real time expert system, which codifies the ICAM system internal and external behavior in its knowledge base [30]. The external plant model represents an oil production facility, which separates oil well fluids into crude oil, sales gas, and water [19].

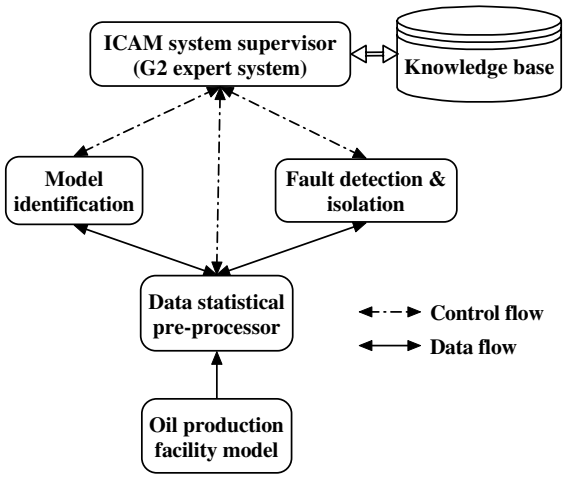

Fig. 3. ICAM system prototype

A. ICAM system prototype simulation

To analyze the performance of the ICAM system prototype in terms of computation and communication utilization, a simple simulation experiment was conducted, as illustrated in figure 4. We tested two agents of the ICAM system prototype for simplicity. The two agents are executed on two PC machines connected to a Windows-based local network. The first agent is an oil production facility model, which was

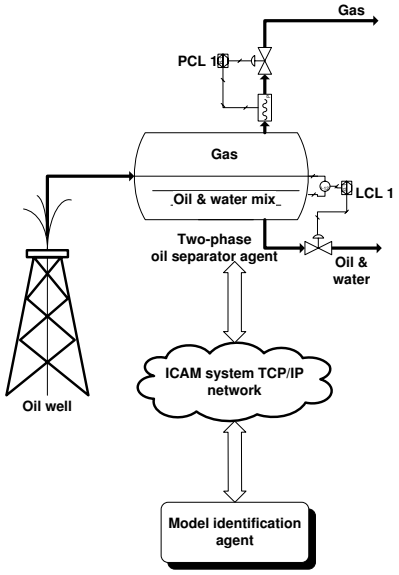

Fig. 4. ICAM system prototype simulation

thoroughly developed as a test-bed for the ICAM system verification and validation [19]. The model consists of 10 states, 5 manipulated variables, 5 controlled variables, and 17 auxiliary measured inputs and outputs (e.g., disturbances, product quality variables, etc....). The second agent in the experiment is a model identification agent, which incorporates the recursive least squares (RLS) with forgetting factor ID technique. A second-order multi-input single-output au-

Profile Summary

Generated 28-Dec-2006 14:49:35 using real time.

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
eufix1yRT	1	360.054 s	313.620 s	
separator>sep_ode	2867	45.919 s	0.251 s	
fzero	2867	45.106 s	6.638 s	
inline.subsref	93309	38.436 s	8.748 s	
inlineeval	93309	29.688 s	29.688 s	
MPIConnect (MEX-function)	2870	0.500 s	0.500 s	

Fig. 5. The two-phase oil separator model agent performance profile

Profile Summary

Generated 28-Dec-2006 14:49:36 using real time.

Function Name	Calls	**Total Time**	Self Time*	Total Time Plot (dark band = self time)
MPIConnect (MEX-function)	2868	359.159 s	359.159 s	
rarx	5736	0.031 s	0.031 s	

Fig. 6. The model identification agent performance profile

toregressive with exogenous input (ARX) model structure was used with a forgetting factor of 0.98. The simulation experiment was conducted in real time with a sampling period of 100 milli-seconds, and a time span of 6 minutes. The exchanged messages between agents included all the 37 process variables of the whole production facility model. We focused on the identification of the first process of the production facility model, which is a two-phase oil separator. The two-phase oil separation process separates light-weight hydrocarbon gases from oil-well fluids. The liquid level of the separator is controlled by manipulating the liquid outflow valve (as indicated by the LCL1 control loop in figure 4). Additionally, the separator pressure is controlled via the gas discharge valve (as indicated by the PCL1 control loop in figure 4). The model ID agent receives data messages from the two-phase separator agent and identifies the separator model recursively in real time. The RLS identification was applied twice every sampling period, in the sense that one output was identified at a time as a function of the two inputs.

B. Simulation results

The performance of each agent in the experiment was measured by using the Profiler functionality in MATLAB, in which the total time of every code line is calculated in real time. Figure 5 illustrates the profile summary of the first agent (i.e., the oil production facility model agent). The first agent used our first-order fixed-step Euler ordinary deferential equation (ODE) solver, which was developed to meet the real time and MPI communication requirements. The most important functions in the table are the ODE solver (i.e., the first line in the table), the separator ODE model (i.e., the second line), and the MPI communication protocol (i.e., the last line). The first observation to be noticed is that the total time of the ODE solver was 360.054 seconds, which equals the total simulation time (i.e., 6 minutes). Only a time of 0.5 second is consumed by the MPI communication task, as indicated by the total time of the MPIConnect function, although it was called 2870 times during the simulation. This verifies the high performance of the developed ICAM agent despite its implementation in Matlab.

The total time of the separator ODE model is 45.919 seconds due to the fact that a nonlinear optimization problem is solved every sampling period [19]. The number of separator ODE model calls is 2867, which is three less than the number of the MPI communication function calls. This is because the MPI communication function had to be called to end the computation task of the second agent, and to initialize and finalize the MPI communication task.

Figure 6 illustrates the performance of the second agent (i.e., the model identification agent). It is very interesting to notice that the communication task in the second agent consumed the largest amount of time (i.e., indicated by the self time of the MPIConnect function). This is due to the global synchronization between the two agents. The recursive autoregressive model identification algorithm (i.e., the rarx function) consumed only 0.031 seconds, although it was called 5736 times. The number of the model identification function calls was twice the number of the MPI communication function calls. This is because only one output of the 2×2 separator process was identified at a time, as explained in the previous section.

The experiment was repeated to study the network activity, as shown in figure 7. The bit transfer rate of the 100 Mbps network spiked up to 395 Kbps during the MPI environment initialization. Then the bit rate settled down to 100 Kbps during the normal operation of the ICAM system prototype. Once the computation and communication tasks at each agent ended up the bit rate decreased to a level of 20 Kbps. The bit rate spiked up to 50 Kbps and then decreased to a level of less than 10 Kbps during the shutdown of the ICAM system prototype.

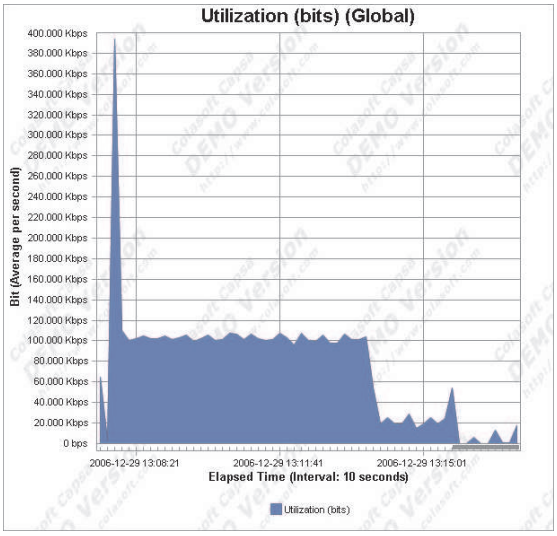

Fig. 7. Network utilization during simulation

Figure 8 shows the identification results of the model ID agent during the experiment, which validates the model ID agent performance in a distributed real time environment. The liquid level and separator pressure set points of the two-phase separator process in the first agent were stepped up by 20% simultaneously at time $t = 30\,sec$. The model ID agent identified the model of the 2×2 two-phase separator process. As indicated by the top plots of figure 8. The estimated liquid level and gas pressure process variables (i.e., the dash-plus trace) exactly match the real-time simulated process variables (i.e., the dashed trace), which are received from the first agent. The bottom plots of figure 8 show the control actions of the two-phase separator during the experiment.

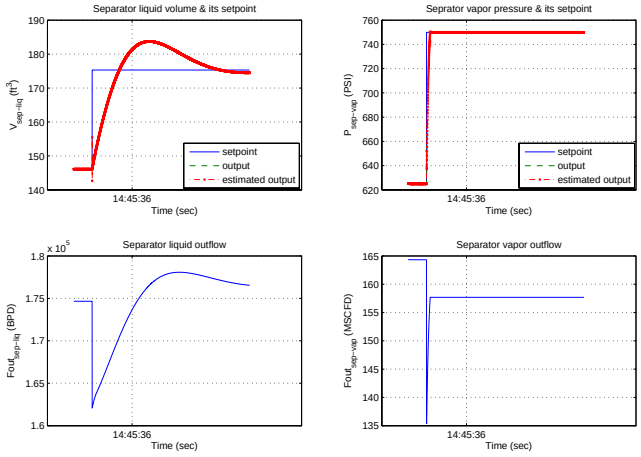

Fig. 8. Model identification agent simulation results

VII. CONCLUSIONS

As part of the PAWS project, we have demonstrated good progress in the design and development of the ICAM system. The ICAM agent structure was designed to achieve autonomy requirements in terms of best communication and computation overlapping scheme. Based on the message passing interface (MPI) communication standard, the communication requirements of the ICAM system were analyzed. In order to guarantee a robust and coherent system performance, the ICAM system's artificial intelligence requirements and tools have been introduced in a separate work [23]. The software implementation of the ICAM agents was discussed. An ICAM system prototype was described and demonstrated. A real time simulation experiment was conducted on two agents of the prototype to analyze its performance in a real time distributed environment. The simulation experiment results demonstrated good performance, which will be further assessed after the comprehensive ICAM system prototype has been built. More importantly, the experiment results supported our requirements analysis and design decisions to use MATLAB and the MPI communication standard to develop and rapidly prototype real time distributed multi-agent systems. We believe that the ICAM system will pave the way to real intelligent multi-agent systems for many applications.

VIII. ACKNOWLEDGEMENT

This project is supported by Atlantic Canada Opportunities Agency (ACOA) under the Atlantic Innovation Fund (AIF) program. The authors gratefully acknowledge that support

and the collaboration of 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 the 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] ——, "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, in *Petroluem 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," in *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] A. F. Sayda and J. H. Taylor, "An intelligent multi agent system for integrated control and asset management of petroleum production facilities," in *The 17th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM)*, Philadelphia, USA, 18-20 June 2007.

[24] M. Omana and J. H. Taylor, "Fault detection and isolation using the generalized parity vector technique in the absence of a mathematical model," in *IEEE Conference on Control Applications (CCA)*, Singapore, 1-3 October 2007.

[25] 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.

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

[27] 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.

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

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

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

[31] 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.

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