A Multi-agent System for Integrated Control and Asset Management of Petroleum Production Facilities - Part 3: Performance Analysis and System Limitations

Atalla F. Sayda and James H. Taylor

Abstract— **This three-part paper thoroughly addresses the design and development of 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], defined its autonomy, communications, and artificial intelligence (AI) requirements [4], [5], and initiated the preliminary design of a simple system prototype [6], we are extending the build of a system prototype and simulate it in real-time to validate its logical behavior in normal and abnormal process situations and analyze its performance. The third-part paper addresses the ICAM system prototype validation in terms of system performance analysis and system behavior during unexpected situations.**

I. INTRODUCTION

As part of the ICAM system prototype development, the third-part paper addresses the system prototype validation and its performance analysis in real time. This would reveal any computational bottlenecks to be rectified in future work. Furthermore, this paper also investigates the limitations of the ICAM system prototype during unexpected situations [1], [7], [2], [8], [9], [3], [10], [11], [12], [13], [4], [14], [5], [15], [6]. To accomplish this, a real-time simulation experiment was designed to analyze the performance of the ICAM system prototype in terms of its logical behavior and its response to the external environment dynamics. The ICAM system prototype is deployed in a Windows 2003 network, which has two nodes (i.e., workstations). The first node has three running agents, namely the pilot plant agent, the model ID agent, and the supervisory agent. The second node has the remaining agents, namely the statistical preprocessing agent and the FDIA agent. The pilot plant simulation model corresponds to figure 1; it consists of 10 states, 5 manipulated variables, 5 controlled variables, and 17 auxiliary measured inputs and outputs (e.g., disturbances, product quality variables, etc.). Ten sensor/actuator faults are embedded in the pilot plant simulation agent to emulate faulty instrumentation in real-world oil production plants, as indicated in table I.

In the real-time simulation experiment, we will apply a bias fault in the three-phase separator water volume sensor and conduct an average system performance, as described in section 2. However, A more rigorous system performance analysis is required to see the big picture, as discussed in section 3. Furthermore, we will apply two different simulation scenarios and discuss the ICAM system prototype behavior during unexpected plant disturbances and situations in sections 4, 5 and 6. Finally we will discuss the system prototype design limitations and how they can be rectified in future work in section 7.

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

Fault number	Instrumentation name
F1	Faulty two-phase liquid volume sensor
F2	Faulty two-phase pressure sensor
F3	Faulty three-phase water volume sensor
F4	Faulty three-phase oil volume sensor
F5	Faulty three-phase pressure sensor
F6	Faulty two-phase separator liquid outflow valve
F7	Faulty two-phase separator gas outflow valve
F8	Faulty three-phase separator water outflow valve
F9	Faulty three-phase separator oil outflow valve
F10	Faulty three-phase separator gas outflow valve

TABLE I
OIL PRODUCTION FACILITY INSTRUMENTATION FAULTS

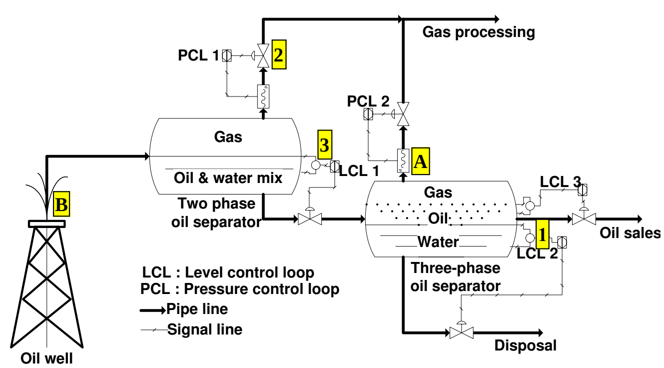

Fig. 1. Oil production facility P&ID

II. ICAM SYSTEM PROTOTYPE AVERAGE PERFORMANCE

Having verified the ICAM system prototype functionality in the second-part paper, it is crucial to analyze its performance to pinpoint any computational bottlenecks and to verify the correctness of the computation/communication overlap (i.e., the correct order of communication and computation tasks in each agent's code). The performance analysis is done during the fault accommodation system mode of the same simulation scenario, which was discussed in the second part of this paper. We did profiling on the code of every reactive agent to determine the average execution cycle of the reactive agent and its tasks. Table II illustrates the profile of the pilot plant agent, whose execution cycle took 108.543 milliseconds.

It is evident that the real-time clock functionality took the biggest execution time slot (i.e., about 70.49%). The evaluation of the oil separator ordinary differential equation (ODE) model took 24.12% of the total execution time, due to the nonlinear problem being solved every sampling period [16]. Raw data storage consumed around 1.77% of the agent's execution cycle. Communicating data to other agents and messages to the supervisory agent did not have a significant effect on the agent's performance, which indicates a good communication/computation overlap.

While computational functionalities dominated the pilot plant agent, data communications with other agents took the largest execution time slot in the statistical preprocessing agent. That is, the raw data reception task took 69.64% of the agent's execution cycle, as demonstrated in table III. This is due to synchronization with other agents during data reception, i.e., waiting. However, this is less significant on the agent performance when sending processed data to other agents (i.e., about 11.99% of the execution time), as specified by the system design requirements [3], [4]. The total execution cycle of this agent was 106.98 milliseconds. It is evident that there is a performance bottleneck in this agent due to raw and processed data storage (around 16.8%). This can be rectified by adding a database agent to the system which stores the different data types across the ICAM system. Again, the communication part with the supervisor had a minimum effect on performance.

Functionality name	Total Time Per Execution Cycle (ms)	% Time
Real time clock	76.512	70.49%
Separator ODE model evaluation	26.18	24.12%
Accommodation data reception	3.387	3.12%
Raw data storage	1.925	1.77%
Raw data sending	0.336	0.31%
Communication with supervisor	0.203	0.18%
Totals	108.543	100%

TABLE II
THE PILOT PLANT AGENT PERFORMANCE PROFILE

When it comes to the model ID agent, the reception of processed data from the statistical preprocessing agent had the biggest effect on performance (i.e., about 99.48% of the agent's execution cycle time). While the execution cycle of this agent took 105.69 milliseconds, communications with the supervisory agent had the least effect on performance, as shown in table IV. The FDIA agent had a similar profile of the model ID agent, in which data communications took 90.89% of the agent's cycle execution time. We do notice here that data storage has a fairly undesirable effect of 13.24% on the FDIA agent performance, as illustrated in table V. Table VI demonstrates the performance of the supervisory agent during the real-time system simulation. The G2 supervisory agent was in an idle state for almost 99.18% of the total simulation time, whereas communications with other agents had almost no impact on the agent's performance, as specified in the design requirements.

Functionality name	Total Time Per Execution Cycle (ms)	% Time
Raw data reception from pilot plant	74.50	69.64%
Processed data sending	12.837	11.99%
Raw data storage	9.451	8.83%
Processed data storage	8.54	7.98%
Outlier removal	1.404	1.31%
Communication with supervisor	0.248	0.23%
Totals	106.98	100%

TABLE III
THE STATISTICAL AGENT PERFORMANCE PROFILE

Functionality name	Total Time Per Execution Cycle (ms)	% Time
Processed data reception	105.15	99.48%
Communication with supervisor	0.54	0.52%
Totals	105.69	100%

TABLE IV
THE MODEL ID AGENT PERFORMANCE PROFILE

Functionality name	Total Time Per Execution Cycle (ms)	% Time
Accommodation data sending	68.601	45.67%
Processed data reception	67.916	45.22%
Processed data storage	7.45	4.96%
FDI variables storage	5.763	3.84%
Communication with supervisor	0.464	0.31%
Totals	150.194	100%

TABLE V
THE FDI AGENT PERFORMANCE PROFILE

III. COMPLETE ICAM SYSTEM PROTOTYPE PERFORMANCE ANALYSIS

Although the ICAM system prototype performance analysis showed good results, the performance analysis was a snapshot done during the fault accommodation mode of the system. It is crucial to conduct a complete system performance analysis throughout the complete life-time of the ICAM system. The ICAM system goes through six

Functionality name	Total Time	% Time
Idle time	1575 s	99.18%
Scheduling time	0.982 s	0.06%
Communication with agents	4.505 s	0.28%
Other functionalities	7.565 s	0.47%
Totals	1588.052 s	100%

TABLE VI
THE SUPERVISORY AGENT PERFORMANCE PROFILE

different modes during its life-cycle, namely, system startup and steady state detection mode, PRBS signal application mode, model identification mode, fault diagnosis mode, fault accommodation mode, and system shutdown mode. A real-time simulation scenario was set up to measure the execution cycles of the different reactive agents and compare it against the ICAM system network activity during the six modes of the ICAM system. We studied scenario $\boxed{3}$ (refer to $\boxed{3}$ in figure 1), i.e., we applied a bias fault in the liquid volume sensor of the two-phase separator to make the system execute the fault diagnosis and accommodation modes.

Figure 2 shows the measured execution cycles of the ICAM system reactive agents, where the overlapped execution cycles traces show remarkable synchronization among the reactive agents during the six system modes, since the agents have nearly the same execution cycle traces. The ICAM system agents start up with an execution cycle of 94 milliseconds, which increases to 95 milliseconds during the steady state detection mode (i.e., mode 1). It is interesting to notice that the execution cycle increases to a level of 110.7 milliseconds during mode 1. The execution cycle of the model ID agent increases to around 92 milliseconds because of the time-consuming plant model estimation task. The processed data MPI channel is closed during this mode and the FDIA agent enters a waiting state till the end of mode 2, as shown in figure 2.

The pilot plant and statistical preprocessing agents continue executing their functionalities at a cycle level of 110.7 milliseconds. The agents' execution cycle decreases to a level of 98 milliseconds and then increases to a level of 109 milliseconds during the fault diagnosis mode (i.e, mode 4) and the beginning of the fault accommodation mode (i.e, mode 5). The execution cycle then increases to a level of 124 milliseconds during the fault accommodation and system shutdown modes. The gradual increases of the execution cycles of the agents are accompanied with matching gradual increases in agents' memory consumption and matching gradual decreases in network activity (i.e., less communications among agents). This interesting phenomenon is attributed to the fact that some agents store their local data in large matrices, whose growing size requires more computational effort and more memory consumption. This reflects on the communications and network activity of the ICAM system as demonstrated by figure 4.

Figure 3 shows the individual execution cycles of the ICAM system reactive agents, which again demonstrates the agents' remarkable synchronization in spite of the semi-autonomous nature of the ICAM system agents. We measured the network activity of the ICAM system prototype in this simulation scenario, as illustrated in figure 4. It is interesting to notice that the communication activity of the agents is a mirror of the computation activity. The gradual decreases in network activity match the gradual increases in computational activity of the agents, as observed in figures 2 and 4. It is also observed that the processed data MPI channel is closed during the model ID mode (i.e., mode 2) because of the time-consuming task of plant model estimation (as indicated by the yellow trace in figure 4). This simulation scenario demonstrated an excellent ICAM system performance in terms of good computation/communications activities overlap, as specified by design requirements. It also highlighted the need to embed a data-base management agent in the ICAM system to relax the execution cycle of the reactive agents, and hence, to improve the ICAM system performance.

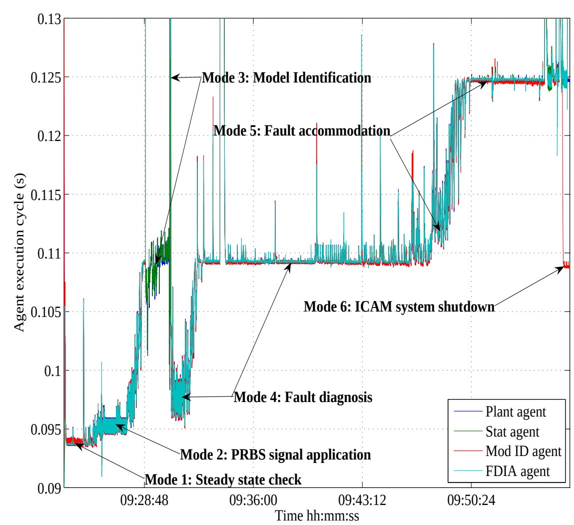

Fig. 2. Execution cycles of ICAM system agents (overlapped)

IV. ICAM System Prototype Limitations

The ICAM system prototype showed excellent logical behavior in response to simple faulty sensors/actuator scenarios. However, the system has limitations that must be identified and analyzed carefully. This will reveal if the system will respond consistently against unexpected disturbances, and show a coherent performance and acceptable logical behavior. Two simulation scenarios have been applied to study the ICAM system prototype limitations, as discussed in the following sections.

V. Scenario A: ICAM system behavior during faults with fast dynamics

To demonstrate the system behavior during fault with fast dynamics, a $+15\%$ bias fault is applied to the three-phase separator pressure sensor (F5; refer to control loop PCL2 sensor fault is almost immediately masked by a quick spike in outflow, allowing the pressure to adjust to the erroneous setpoint so fast that correct isolation is impossible.

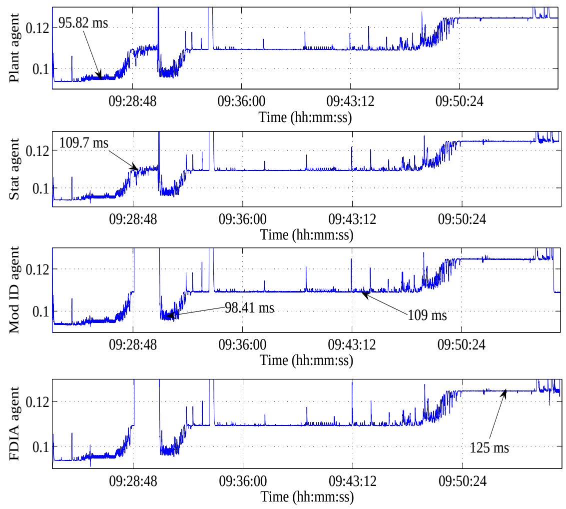

Fig. 3. Execution cycles of ICAM system agents (non overlapped)

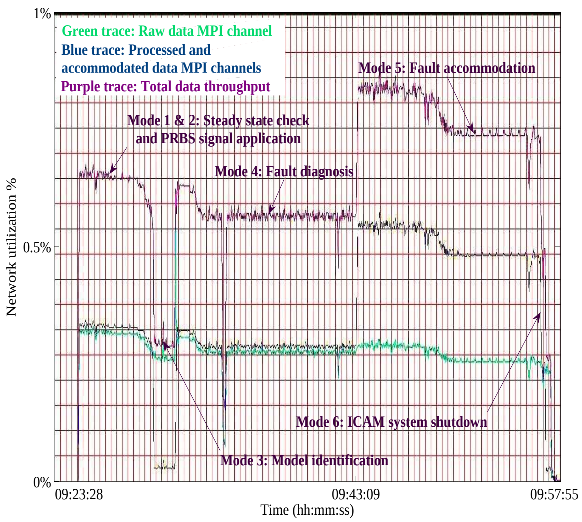

Fig. 4. ICAM system prototype network activity

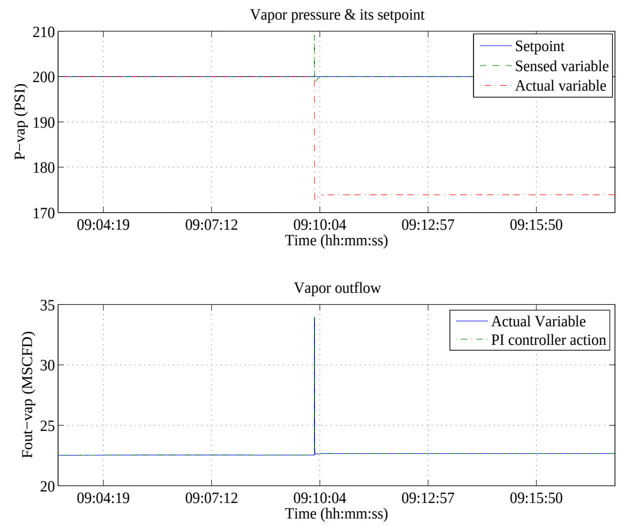

Fig. 5. Limitation scenario A: Three-phase separator pressure logged by the FDIA agent

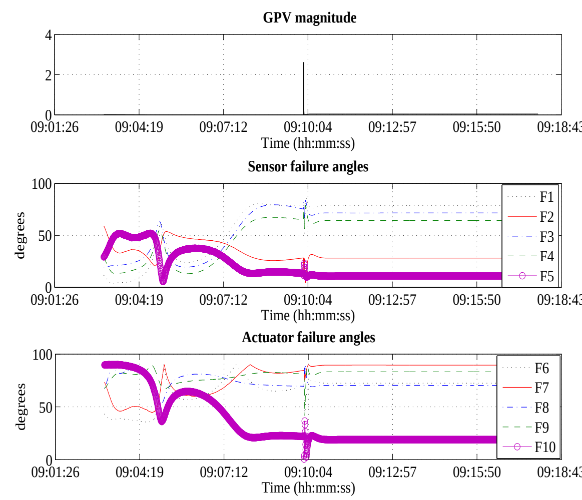

Fig. 6. Limitation scenario A: FDIA agent diagnostic signals

in figure 1) at time T_{fault} = 09:09:55. Figure 5 shows the measured and actual pressure of the three-phase separator along with its associated gas outflow. The figure obviously shows that there is a significant mismatch between the measured and actual pressure. The FDIA agent successfully detects a fault at time T_{fault} = 09:09:55 (refer to table VII), as the GPV magnitude spikes up sharply, as shown in the top plot of figure 6. The lowest GPV angle compared to other angles corresponds to fault F10, where it decreases sharply for a short time period, as shown in the bottom plot of figure 6. Accordingly the FDIA agent internal logic declares that a fault F10 has been isolated, as shown in figure 7. Interestingly enough, the isolation decision lasts for a very short period of time. Fault F10 corresponds to a fault in the three-phase separator gas outflow valve (refer to table I), which is in the same control loop as F5. The pressure

The FDIA agent sends the fault information to the supervisory agent to reason about this abnormal plant situation, as shown in the FDIA agent supervisory frame in table VII. The supervisory agent decides not to activate the fault accommodation task as the fault is identified an actuator fault (refer to the accommodation attributes in table VII). As a result, the fault accommodation parameters have a value of zero and the fault size and sign attributes have no facts, as the identified actuator fault size can not be estimated by the FDIA agent. It is very evident that something went wrong during this simulation scenario, as the system isolated the wrong fault (i.e., fault F10 instead of fault F5).

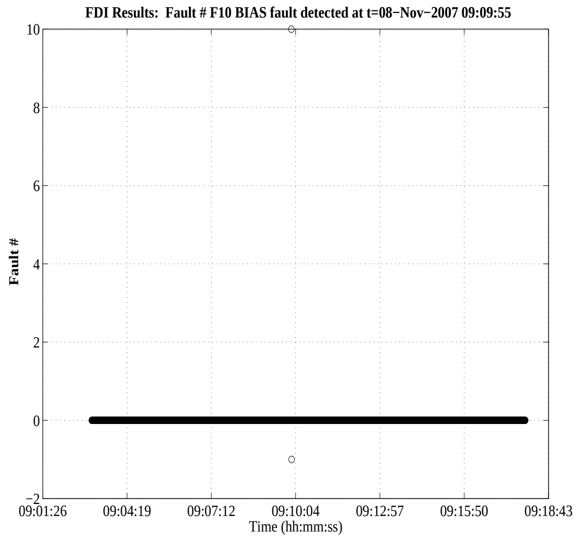

Fig. 7. Limitation scenario A: FDIA agent fault display

Names	FDIA-OBJECT
Rank	3
State	simulate
Mpi comm	comm-world
Mpi comm size	4
G2 link status	connected
Mpi link status	connected
Decision	diagnose-faults
Simulation status	on
Mpi channel decision	no-decision
Model status	model-is-identified
Fdi design status	fdi-filter-is-designed
Fault	f10
Fault sign	none
Fault size	NaN
Fault type	bias
Fault time	"08-Nov-2007 09:09:55"
Accommodation status	none
Recursive fault size estimation status	none
Acknowledge fault	off

TABLE VII

LIMITATION SCENARIO A: FDIA AGENT SUPERVISORY FRAME

To analyze this situation, we compare the data record of the three-phase separator pressure measurement logged at the FDIA agent (refer to figure 5) and the same data record logged at the pilot plant agent (refer to figure 8). The comparison reveals that there is a significant difference between the two pressure measurement data records at the fault application instant. The pressure measurement logged at the FDIA agent shows that the pressure spikes up to $P = 209$ PSI compared to a value of $P = 230$ PSI logged at the pilot plant agent during fault application. The same difference can be noticed in the three-phase separator gas outflow measurement data records. This measurement mismatch is due to the fast dynamics nature of the three-phase separator pressure, which caused the outlier removing task in the statistical preprocessing agent to clip the pressure and gas outflow measurements before being sent to the FDIA agent. This led to the wrong fault isolation decision made by the FDIA agent, and the wrong decisions made by the supervisory agent accordingly. This highlights the importance of embedding a safety net in the ICAM system prototype to compensate for the limitations of the system agents; for example, if the statistical preprocessing agent notified the other agents whenever it clipped a measurement they could be able to make allowances for that fact. Refining the statistical preprocessing agent so that pressure spikes were not treated as outliers would also be effective.

VI. SCENARIO B: OIL-WELL PRODUCTION DECREASE

The productivity of offshore oil wells and fields may decrease with time and demand, which leads to a decrease in the oil flow to the production facility. In order to analyze the impact of such change on the logical behavior of the ICAM system prototype, we introduce a 20% sudden decrease in the oil component of the oil-well incoming flow at time $T_{dist} =$ 14:58:00 (refer to the symbol $\boxed{\text{B}}$ in figure 1).

Figure 9 shows the two-phase liquid volume measurement logged at the FDIA agent, where the disturbance effect on the liquid volume is corrected by the PI controller by adjusting the liquid outflow valve accordingly. The disturbance is rejected in a time period of about seven minutes, during which the GPV magnitude increases and stays at value of 0.4, as shown in the top plot of figure 10. Interestingly enough, none of the GPV angles go to a low level except for the angle of fault F7 for a very short time period, as illustrated in the middle and bottom plots of figure 10. The local decision making logic of the FDIA agent declares that a fault F7 is isolated for a short time period, as shown in figure 11. The decision of the FDIA agent then takes a value of -2, which corresponds to an undefined fault decision

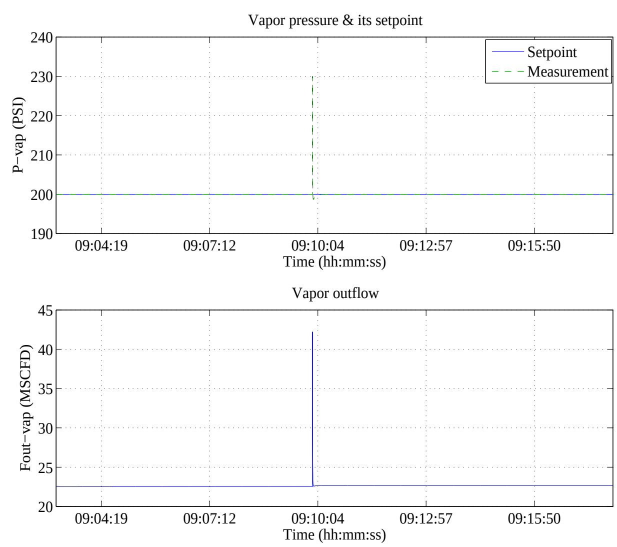

Fig. 8. Limitation scenario A: Three-phase separator pressure logged by the pilot plant agent

during steady state. The FDIA decision then changes to -1 for a longer time period, which represents an undefined fault during transient. This strange behavior can also be noticed in the GPV magnitude (refer to the top plot of figure 10), where the GPV magnitude seems to reach a steady state of about 0.2 for a very short time period and then increases for a longer time period before it is in a true steady state. Finally, the FDIA fault isolation decision settles down on a fault F7 (i.e., the last detected fault) till the end of the simulation scenario, although the lowest GPV angle is the one associated with fault F1.

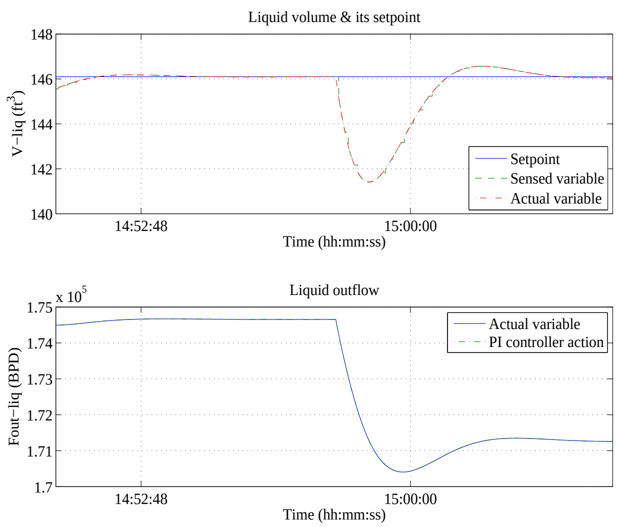

Fig. 9. Limitation scenario B: Two-phase separator liquid volume logged by the FDIA agent

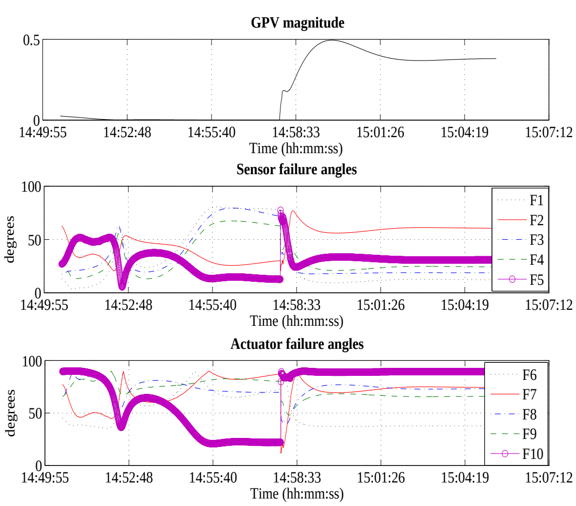

Fig. 10. Limitation scenario B: FDIA agent diagnostic signals

In order to verify the FDIA fault isolation decision, we examine the FDIA agent supervisory frame, depicted in table VIII. It is evident that the FDIA agent isolates a fault F7 of type ramp at time T_{fault} = 14:58:04. Fault F7 corresponds

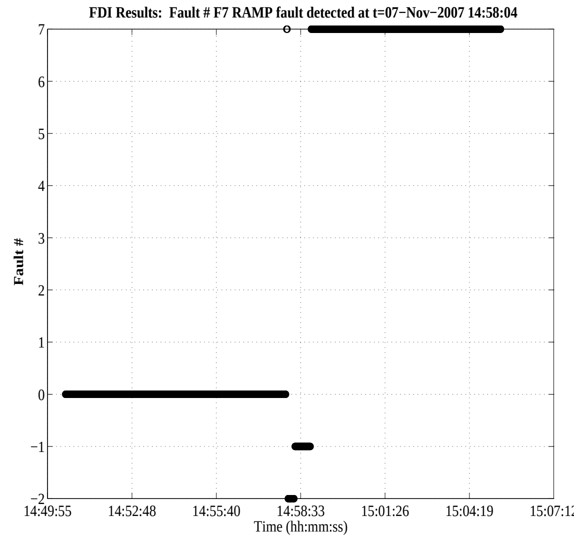

Fig. 11. Limitation scenario B: FDIA agent fault display

to a faulty two-phase gas outflow valve (refer to table I). The supervisory agent reasons about this situation and decides that no fault accommodation action should be taken, as indicated by the accommodation attributes in table VIII. The fault sign and size attributes provide no new facts, as the identified fault is an actuator fault, whose size can not estimated by the FDIA agent. The FDIA agent can only estimate sensor faults by design.

To further analyze the results, we plot the two-phase separator pressure measurement and its associated gas outflow data records logged at the FDIA agent, as shown by figure 12. The gas pressure and outflow measurements show the effect of the oil-well incoming flow disturbance, which is rejected by the PI control loop PCL1 (refer to figure 1). Furthermore, there is no mismatch between the measured and actual data records for both process variables. To add to the situation complexity, the FDIA agent was not designed to isolate ramp actuator faults. Yet, it did declare that a ramp actuator (i.e., gas outflow valve) fault has occurred. This complex simulation situation, in which the FDIA agent generated confusing decisions, is attributed to the fact that the FDIA agent was not designed to decouple disturbances from sensor/actuator faults. This limitation is due to a lack of an analytic model that includes disturbance inputs; studies have demonstrated that known disturbances can be decoupled so they do not interfere with FDI [11]. Again, the necessity of embedding limitations of the ICAM system reactive agents in the knowledge base of the supervisory agent becomes crucial to guarantee robust and logically coherent system behavior.

VII. System Limitations, Design Challenges, and Future Work

Designing an intelligent multi-agent system is a very challenging task, as all agents are distributed and semi-autonomous. We faced several design challenges which resulted in limited system capabilities. Some of these design challenges and the future recommendations for solving them

Names	FDIA-OBJECT
Rank	3
State	simulate
Mpi comm	comm-world
Mpi comm size	4
G2 link status	connected
Mpi link status	connected
Decision	diagnose-faults
Simulation status	on
Mpi channel decision	no-decision
Model status	model-is-identified
Fdi design status	fdi-filter-is-designed
Fault	f7
Fault sign	none
Fault size	NaN
Fault type	ramp
Fault time	"07-Nov-2007 14:58:04"
Accommodation status	acc-not-possible
Recursive fault size estimation status	recalculation-not-required
Acknowledge fault	off

TABLE VIII
LIMITATION SCENARIO B: FDIA AGENT SUPERVISORY FRAME

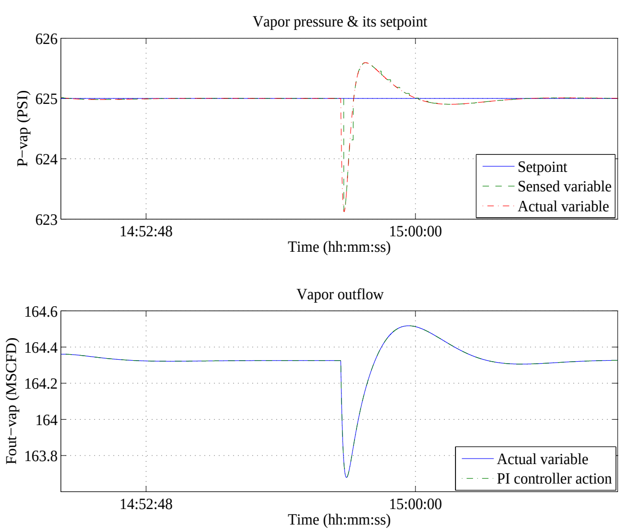

Fig. 12. Limitation scenario B: Two-phase separator pressure logged by the FDIA agent

are suggested in the following points:

- Although we proposed the hierarchical colored petri nets approach to design the internal logic of the ICAM system reactive agents in our development plan [3], we did design the agents' internal logic in an *ad hoc* manner. We faced some difficulties during the design stage of the ICAM system prototype, as more functionalities were added. For example, the ICAM system crashed during early simulation runs due to communication deadlocks, in which two agents were trying to send messages to each other simultaneously. The problem was solved by imposing conditions on communicating agents to prevent such deadlocks. Future designs should use the colored petri net approach to verify the logical behavior of the ICAM system and its agents in different scenarios.
- Computation/communication coordination was another design problem, in which computation and communication code blocks were not ordered correctly in the agent code. For example, we combined the process model estimation (computation task) and sending the estimated model to other agents (communication task) into one task in the model ID agent, which proved to be a design flaw. Model estimation took a long time (i.e., over one minute), during which other agents were locked waiting for the estimated model due to synchronization failure. The problem was solved by separating the one functionality into two separate computation and communication functionalities (i.e., separate agent states) and modifying other agents accordingly. Although some design flaws had to be corrected, the ICAM system prototype acted as a set of distributed stochastic colored petri nets during real-time simulation. This implies that a careful agent design should be done along with a thorough system logical behavior analysis. Future design plans would take the stochastic nature of the system and time into account to guarantee robust performance.
- The plant data characteristics also had a major impact on the ICAM system performance. For example, the ICAM system prototype is not robust against noisy data due to the design of the data differentiation-based steady state detection algorithm. Likewise, the general parity vector (GPV) based FDIA algorithm is not robust to noise, which significantly affects the fault isolation task in moderate to high noisy data situation. We suggest embedding algorithms that are more robust to noise to cope with real-world industrial plants and their noisy measurements.
- Detection and isolation of fast dynamics faults (e.g., faulty gas pressure sensor) is another limitation of the ICAM system prototype. The outlier removal algorithm in the statistical processing agent treats fast dynamics faults as outliers, which changes the nature of processed data sent to the FDIA agent. Data filtering also may change the data characteristic, which may have an impact on the system performance. In addition, the system logical behavior was unpredictable and inconsistent in response to disturbances in process variables. So we suggest developing a better safety net, in which the knowledge of agents' limitations is embedded in the rule base of the supervisory agent. This allows the system to have a better reasoning ability and robust performance during undefined and unpredictable plant situations.
- In order to address the complete asset management solution in process plants, several agents have to be embedded in the ICAM system prototype to manage the process plants during normal situations. An optimization agent is essential to generate optimal material recipes and process variable set-points to guarantee higher product quality. Planning and scheduling agents are also essential to schedule operation plans in accor-

dance with long term production plans. Furthermore, the addition of a real-time database management agent is vital for both high system performance and future scalability. Finally, a graphical user interface (GUI) agent must be added to the ICAM system to meet process operator interaction requirements.

- The incorporation of domain knowledge would definitely improve the performance of the system. Such knowledge is represented by the topology of the industrial plant and its operation procedure in different situations such as startup, normal operation, and shutdown. This knowledge would be better utilized if a learning agent were embedded to deal with new situations in the plant and the internal behavior of the ICAM system itself.
- During abnormal situations hundreds of alarms are initiated, leading to alarm flooding. This results in the operator missing important alarms. Proper asset management requires proper alarm and event management techniques in addition to good operator decision support. The incorporation of alarm management techniques that can dynamically prioritize important alarms and suppress unnecessary alarms would definitely enhance the ICAM system performance. The alarm management agent would interact with the FDIA agent to better identify the most important alarms that have to be dealt with.

As can be appreciated, those enhancements will require years of additional research and development.

VIII. CONCLUSIONS

A real-time simulation experiment was conducted to validate the ICAM system prototype in terms of its performance and logical behavior in real time. The code of the four reactive agents was profiled to detect any computational bottlenecks. To investigate the ICAM system prototype performance during the different system modes, a rigorous system performance analysis was conducted. Furthermore, two different simulation scenarios were tested to investigate the ICAM system prototype behavior during unexpected plant disturbances and situations. As a result, the designed ICAM system prototype showed some performance degradation and some limitations, which were discussed in details. We have demonstrated good progress in the design and development of the ICAM system prototype in this three-part paper. Although the ICAM system design, verification, and validation tasks proved to be complex, we believe that the ICAM system prototype will pave the way to real intelligent multi-agent systems for many applications.

IX. 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), 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 second 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, pp. 1397–1404.

[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, pp. 1212–1219.

[4] ——, "An intelligent multi agent system for integrated control and asset management of petroleum production facilities," in *In Proc. of The 17th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM)*, Philadelphia, USA, 18-20 June 2007, pp. 851–858.

[5] ——, "Toward a practical multi-agent system for integrated control and asset management of petroleum production facilities," in *IEEE International Symposium on Intelligent Control (ISIC)*, Singapore, 1–3 October 2007.

[6] J. H. Taylor and A. F. Sayda, "Prototype design of a multi-agent system for integrated control and asset management of petroleum production facilities," in *submitted to the American Control Conference (ACC)*, Seattle, Washington, June 11–13 2008.

[7] 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.

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

[9] 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.

[10] E. Durfee, V. R. Lesser, and D. D. Corkill, "Trends in cooperative distributed problem solving," *IEEE Transactions on Knowledge and Data Engineering*, vol. 1, no. 1, pp. 63–83, 1989.

[11] M. Omana and J. H. Taylor, "Enhanced sensor/actuator resolution and robustness analysis for FDI using the extended generalized parity vector technique," in *Proc. of American Control Conference*. Minneapolis, Minn.: IEEE, 14-16 June 2006, pp. 2560–2566.

[12] J. H. Taylor and M. Laylabadi, "A novel adaptive nonlinear dynamic data reconciliation and gross error detection method," in *Proc. of IEEE Conference on Control Applications*. Munich, Germany: IEEE, October 4-6 2006, pp. 1783–1788.

[13] 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: IFAC, May 17-19 2006.

[14] 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.

[15] J. H. Taylor and M. Omana, "Fault detection, isolation and accommodation using the generalized parity vector technique," in *submitted to the IFAC World Congress*, Seoul, Korea, July 6–11 2008.

[16] 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.