Software Agents for Process Monitoring and Notification

Larry Bunch*, Maggie Breedy*, Jeffrey M. Bradshaw*, Marco Carvalho*, Niranjan Suri*, Andrzej Uszok*, Jack Hansen*, Michal Pechoucek**, & Vladimir Marik**

*Institute for Human and Machine Cognition
40 South Alcaniz St., Pensacola, FL 32502 USA
+1 (850) 202-4462
{lbunch, mbreedy, jbradshaw, mcarvalho, nsuri, auszok, jhansen}@ihmc.us

**Gerstner Laboratory, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
(+420-2) 2435 7421
{pechouc, marik}@labe.felk.cvut.cz

ABSTRACT
Safety and efficiency are primary concerns in chemical processing facilities, though the complexity of many such systems often makes it difficult for operators to detect abnormal conditions before they compromise throughput or become hazardous. In this paper, we report initial results from the application of multi-agent systems to monitor complex chemical processes and flexibly and appropriately notify key plant personnel about off-nominal conditions.

Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *multiagent systems, intelligent agents.*
I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods – *representation languages, semantic networks.*
J.2 [**Computer Applications**]: Physical Sciences and Engineering – *chemistry.*

General Terms
Algorithms, Measurement, Design, Reliability, Human Factors, Theory.

Keywords
Agent, chemical process, monitoring, notification, policy, ontology, KAoS, DAML, OWL, KARMEN, FlexFeed.

1. INTRODUCTION
In 2002 NASA initiated a research program to develop and test improved hydrogen production methods for use in rocket propulsion. One element of the program addresses improved plant safety and control through enhanced anomaly detection, diagnostic, and notification methods. The Institute for Human and Machine Cognition (IHMC) has been funded to investigate the potential utility of software agents for process monitoring and multimodal notification of plant personnel concerning off-nominal operating conditions. Two additional elements of the project are reported elsewhere: 1. the use of software agents for alarm root cause diagnosis [11]; and 2. the use of machine learning techniques to automatically generate the means of monitoring plant status, detecting anomalies, and locating their causes [9].

SAC'04, March 14-17, 2004, Nicosia, Cyprus

2. MOTIVATING REQUIREMENTS
Process control operators can be responsible for monitoring hundreds of control loops with thousands of measurements within a plant area. Operators by necessity depend on pre-defined process alarms to detect abnormal conditions. Early detection of abnormal conditions through automated proactive monitoring can help prevent unscheduled shutdowns, reduce production loss and equipment maintenance frequency, and improve operator effectiveness at maintaining efficiency and identifying faults [2, 10]. To these ends, we are developing the KARMEN (KAoS Reactive Monitoring and Event Notification) multi-agent system for monitoring combinations of process conditions that together are abnormal but are individually within normal range (no alarms), identifying failed components when many interdependent components enter alarm states simultaneously (flood of alarms), and enabling both operator and non-operator personnel to dynamically and proactively monitor aspects of the process. We expect that additional capabilities will be added to the system in future phases of the project.

A unique aspect of the research is our effort to support the work of operators and supervisory personnel who may sometimes be remote and only available for asynchronous communication [17]. The ultimate goal would be to allow concerned personnel to remain apprised of important aspects of the plant situation without continuous monitoring. They should be able to receive notifications about specific events of concern remotely on their preferred device (e.g., cell phone, pager, PDA, computer) and with an appropriate level of urgency.

3. SYSTEM COMPONENTS
This section describes the architecture of the KARMEN system and the underlying tools and services including the chemical process simulation, the KAoS policy and domain services, the

NOMADS agent execution environment, and the FlexFeed agent messaging framework.

3.1 Virtual Plant

A schematic of the chemical process simulation and the related agent applications is shown in Figure 1. The simulation environment, or virtual plant, employed for development and testing has been created using a dynamic process simulation controlled by a commercial distributed control system (DCS).

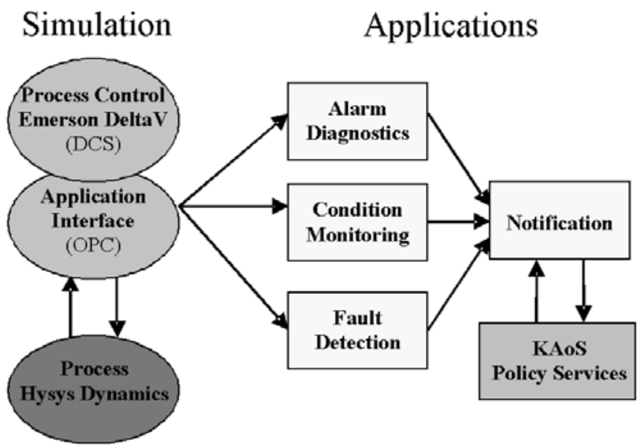

Figure 1. Schematic of the simulation environment and software agent systems.

Aspen Tech's Hysys Dynamics [1] runs the high-fidelity simulation of a Benzene-Toluene separation process which has sufficient complexity and a representative sample of final elements (e.g. valves, pumps, tanks, heat exchangers, sensors) for our implementation. Emerson Process Management's DeltaV Distributed Control System (DCS) [8] controls the process and presents the real-time operator interface that displays measurements and alarms. Companies such as Solutia, Inc. currently use DeltaV to control chemical processing facilities including a steam methane reformer for hydrogen production. Our system obtains all process data from DeltaV through its industry standard OLE for Process Control (OPC) interface [16]. As also depicted in Figure 1, the condition monitoring and notification reported here are part of a collaborative effort addressing abnormal situation management that also includes agent-based alarm root cause diagnosis and the application of machine learning techniques for system-wide fault detection.

3.2 KAoS Policy and Domain Services

KAoS is a collection of componentized agent services compatible with several popular software agent frameworks [3, 4, 13, 19]. KAoS services have also been adapted to run in grid computing (http://www.gridforum.org/) and Web services (http://www.w3.org/2002/ws/) environments as well. In the context of our current implementation, KAoS services are used to define, manage, deconflict, and enforce policies that govern agent notification to plant personnel about process conditions.

The KAoS Policy Ontologies (KPO; http://ontology.coginst.uwf.edu/) are represented in OWL, a declarative, description-logic based markup language (http://www.daml.org/).[1] KAoS relies on an integrated theorem prover (JTP; http://www.ksl.stanford.edu/software/jtp/) along with KAoS-specific extensions to support representation and reasoning about policies.

The current version of KPO defines basic ontologies for actions, actors, groups, places, various entities related to actions (e.g., computing resources), and policies. There are currently over 100 classes defined in the basic ontologies. It is expected that for a given application, developers will further extend KPO with platform-specific and domain-specific classes. As the application runs, classes and individuals corresponding to new policies and instances of application entities are also transparently added and deleted as needed.

The policy ontology distinguishes between authorizations (i.e., constraints that permit or forbid some action) and obligations (i.e., constraints that require some action to be performed, or else serve to waive such a requirement) [7]. A policy is represented as an OWL instance of the appropriate policy type with associated values for properties: priority, update time stamp and a site of enforcement. The most important property value is the name of a controlled action class. In most cases a new action class is built automatically whenever a policy is defined. Through various property restrictions, a given policy can be variously scoped, for example, either to individual agents, to agents of a given class, to agents belonging to a particular group, or to agents running in a given physical place or computational environment. Additional aspects of the action context can be precisely described by restricting values of its properties.

The KAoS Policy Administration Tool (KPAT) provides a graphical user interface for specifying and modifying policies and domains.[2] In addition, KPAT can be used to browse and load ontologies and to deconflict newly defined policies. As policies, domains, and application entities are defined using KPAT, the appropriate OWL representations are generated automatically in the background and asserted into or retracted from the system, insulating the user from having to know OWL or any other policy language. A generic OWL policy editor may be used for this purpose. Specialized policy templates can also be defined to allow various classes of policy definitions to be defined as high-level domain-specific abstractions. A rich set of queries is also available through KPAT or through programmatic interfaces.

Groups of people, agents, and resources are structured into domains and sub-domains to facilitate policy administration. Domains may be nested indefinitely and, depending on whether policy allows, entities may become members of more than one domain at a time.

3.3 FlexFeed Framework

FlexFeed is an agent-based framework designed to provide efficient data distribution in dynamic network environments under strict resource and policy constraints [6]. The main goal of the framework is to abstract all policy and communication requirements from higher level applications.

[1] Because the OWL specification was not finalized at the time this research was performed, we used its predecessor, DAML.

[2] Policies can also be defined, analyzed, or modified programmatically by trusted software components.

In the context of KARMEN, FlexFeed is used as a messaging framework, supporting the enforcement of communication and information release policies between agents. The transport mechanism, message distribution, and filtering are each handled at the framework level, hiding these implementation details from the data producers and consumers. This architecture allows the framework to transparently customize the routing and transformation data streams while abstracting from the agent the tasks associated with the protocol selection, policies, and load balance. Multiple communication protocols and lookup services can coexist in the network and FlexFeed will determine what protocols to use in order to distribute messages between any two nodes. In our current implementation, each node in the network runs the NOMADS Spring execution environment [18]. Any agent running on top of Spring can obtain an instance of the FlexFeedManager object to access the FlexFeed API. To interact with other agents via the framework, an agent must classify itself either as a sink node (one that will potentially work as a client or data sink), or as a source node (sensor), or both (relay or transformation node).

Once registered with FlexFeed, messaging nodes can interact directly through message passing, and listener interfaces. All the routing and policy constraints are enforced by the framework, and can be changed and distributed at run-time.

4. AGENT SYSTEM CONFIGURATION

The capability of software agents to be launched on a network by individual users to perform specified tasks is highly relevant to condition monitoring of modern, complex systems involving a multiplicity of sensors and processors interconnected by hard-wired or wireless interconnections [5, 12, 14]. In the current work we have implemented the KARMEN multi-agent system that monitors specific combinations of process conditions of interest to individual plant operators, supervisors, and other personnel and notify them through modes the individuals select and in accordance with corporate policy and personal preferences.

Figure 2 shows the agent system configuration for such aggregate condition monitoring and multimodal notification. In this system, each Condition Monitor agent is configured and deployed by the user to evaluate a set of conditions over one or more components. The Condition Monitor agent subscribes to one or more Component Agents that access the real-time operational data. When the set of monitored conditions transition between unsatisfied and satisfied, the Condition Monitor agent informs the Notification agent. The Notification agent combines the qualities of the monitored event with the current disposition of the plant personnel, and then uses this data to query KAoS policy services for the appropriate notification modes, salience, and additional recipients.

Each process component in the control system is represented by a Component agent in KARMEN that has real-time access to all operational data about that component through OPC. These Component agents support conditional subscriptions to individual parameters compared to a scalar value and for a given duration. For example, one can subscribe to a particular valve's flow rate dropping below 4.2 kpph for 30 seconds. Such subscriptions are similar to the exceedance alarms supported by the DCS, but KARMEN monitored conditions can be defined dynamically at runtime and the results are published to subscribing agents rather than the DCS operator interface as the conditions obtain and abate. The communication between the Component Agent and its subscribers is performed using the FlexFeed framework described in section 3.2. For efficiency, Component Agents can consolidate multiple subscriptions to a single measurement. This means that a component measure can be used in multiple monitoring cases while requiring only a single feed of measurement data from the DCS. Also for efficiency reasons, Component agents are expected to be deployed close to the source of process data, in this case the DCS but potentially on the component itself.

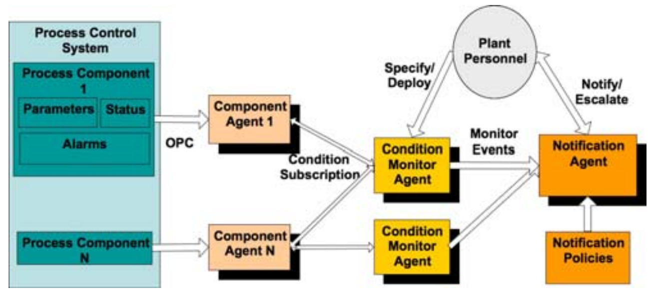

Figure 2. System configuration for the Multi-Agent process condition monitoring and notification software.

Condition Monitor agents are configured and deployed by the users to evaluate logical expressions of conditions across multiple components. When the user launches a Condition Monitor agent, it "subscribes" with the Component agents to receive continual updates on relevant sets of atomic conditions that can trigger notifications. Each time a Component Agent informs the Condition Monitor that one of its conditions has changed, the Condition Monitor agent evaluates its logical expression to determine whether or not the aggregate condition is satisfied. Figure 3 shows the user interface for monitor agent configuration and deployment. The lower-left of Figure 3 depicts how the user browses the hierarchical tree of components and associated parameters available in the DCS. For each selected component parameter in the hierarchy, the user assigns a comparison operator (greater than, less than, equal, etc.), provides a scalar value to which the parameter value will be compared, and optionally provides a duration which the condition must remain true before it is considered satisfied. Each such atomic condition is added to a logical expression using the and/or radio buttons. The evaluation of this expression is the aggregate condition that the agent is monitoring; notification is generated when the entire expression evaluates to true. As shown on the right in Figure 3, the condition monitoring agent is assigned basic notification information including severity (Critical, Warning, Advisory, Log) and an initial target user or role to notify when the conditions are satisfied.

Notifications are generated as the monitored conditions obtain and abate. The mode, salience, and recipients of each notification are governed by KAoS policies representing organizational requirements and personal preferences. Notification modes may include E-mail, Instant Message, Pager, and Operator Displays. Notification policies typically cover such factors as event type, severity, the recipient's organizational role and presence, and the plant area in which the event occurred. For example, a policy might be to page an onsite Field Operator immediately when a critical H2 Plant monitored condition is satisfied and the Process Engineer is unavailable.

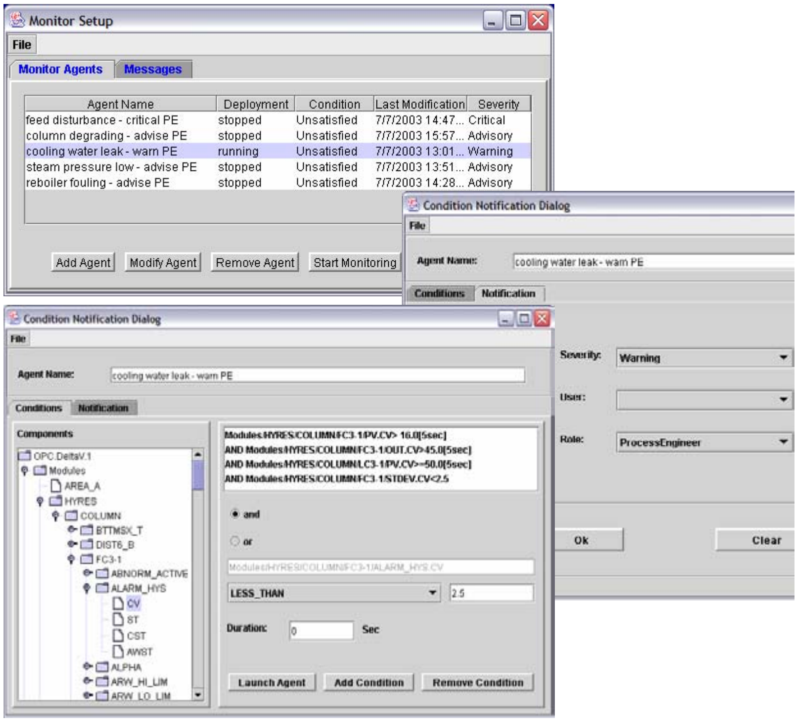

Figure 3: The Monitoring agents configuration application showing clockwise: the agent management and deployment screen, the notification configuration screen, and the monitoring configuration screen.

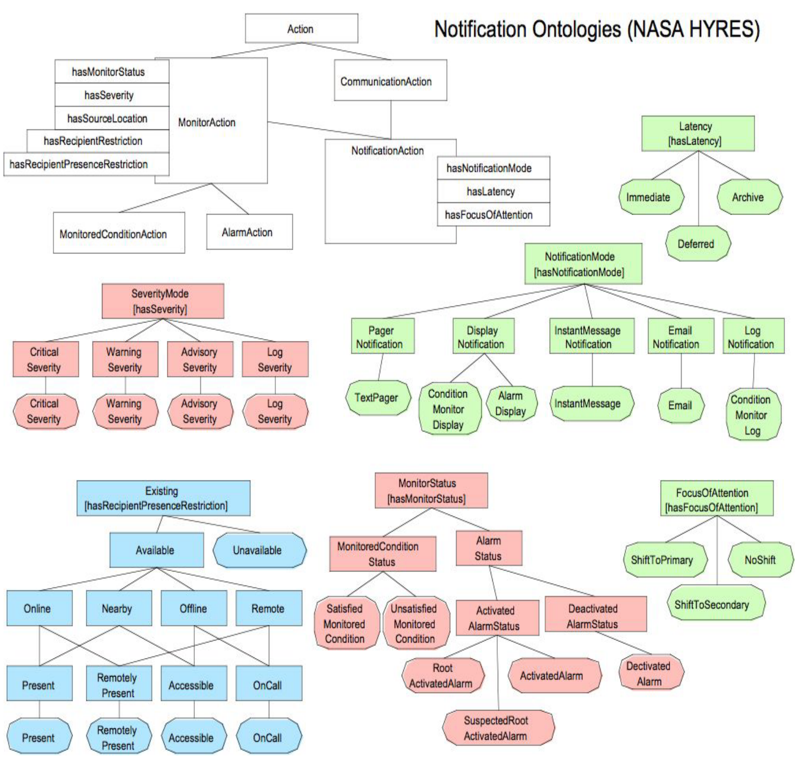

Figure 4: Simplified view of the DAML-based notification ontologies for monitored event characteristics, user presence, and notification directives.

4.1 Policy-Directed Notification

To date most work on complex plant condition monitoring has focused on improving methods of identifying faults [5, 14], with little attention given to the equally important issue of notification of the relevant personnel when off-nominal plant operation occurs. Software agents offer the capability to monitor specific combinations of process conditions and send appropriate notifications as the conditions change.

The notification can occur based on user-specified aggregates and in accordance with the users preferred mode of notification. This contrasts with current process control and monitoring systems which are generally confined to exceedance alarms, control operator displays, and perhaps other more sophisticated information from post-analysis of data collected.

Notifications are generated as the monitored conditions obtain and abate. The default behavior of the Notification agent is to display messages in the Monitor application shown in Figure 3. All other notification actions are governed by KAoS policies representing organizational requirements and personal preferences. Each policy obliges the notification agent to take certain actions based on the qualities of the monitored event and the current disposition of the plant personnel. We have developed a set of initial ontologies depicted in figure 4 for notification that draws heavily on the work of Schrekenghost and colleagues [17]. The current event characteristics that can trigger a policy include the event type (satisfied/unsatisfied condition, activated/deactivated alarm: see MonitorStatus in figure 4), the assigned event severity (critical, warning, advisory, log), and the plant area in which the event occurred based on the component hierarchy defined in the DCS. The user characteristics that can trigger a policy include the user's organizational role (operator, process engineer, area manager, etc.) and the user's current physical and computational presence (nearby/remote, online/offline: see Existing in figure 4). The qualities of the notification action that policies can oblige include the mode, latency, and focus of attention. Notification modes currently include e-mail, instant message, pager, operator displays, and the IHMC Monitor application. The latency controls how quickly the user is notified (immediate, deferred, archive). The focus of attention controls how forcefully the user's attention is obtained and depends on the features available in each notification mode (e.g. instant message chat session that interrupts the user vs. a queued message in the background).

The notification obligation policies are created using the KAoS Policy Administration Tool (KPAT) shown in figure 5. The attributes of the policy are from the ontological concepts shown in figure 4. Multiple policies can apply to a single event such as using the pager mode for critical events, using the instant message mode for critical events when the user is online, and using the primary focus of attention for

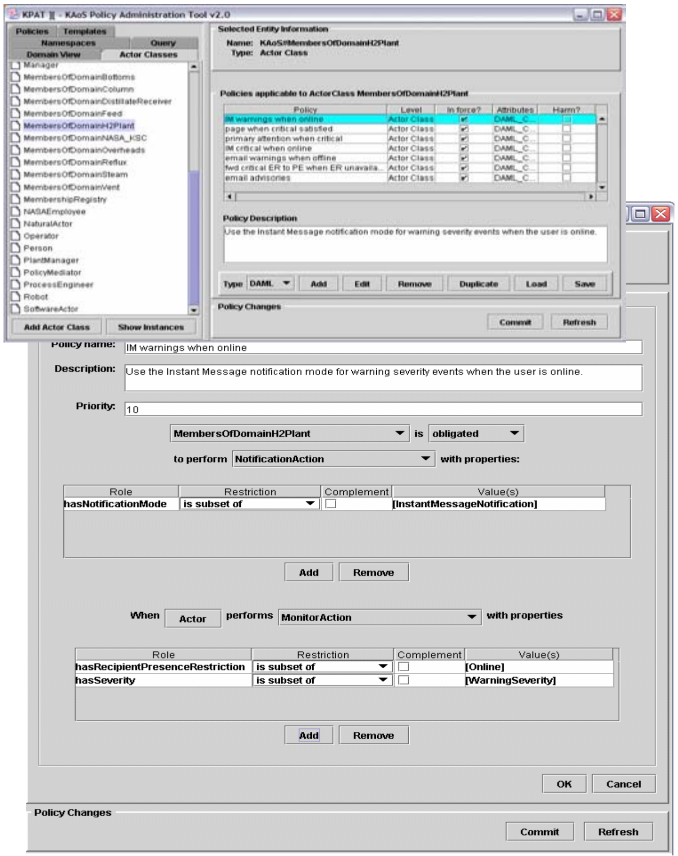

Figure 5: The KAoS policy administration tool (KPAT) shown with a set of notification policies (top) and the configuration of one notification policy (bottom).

critical events. Each policy is assigned a priority, with organizational policies having higher priority than personal preferences. KAoS uses the priority to resolve policy conflicts thereby enforcing organizational policies over personal ones.

5. RELATED WORK

Several research groups have investigated the application of multi-agent systems to process control, fault analysis, and planning. Other AI techniques have also been applied to abnormal situation management and multi-modal notification has been studied for human-agent teamwork. The novel aspects of the KARMEN system include targeting notification through organizational and personal policies as well as the dynamic allocation of user-configured monitoring agents at runtime. Researchers at Czech Technical University's Gerstner Laboratory have applied multi-agent systems in the manufacturing industry including the prototype for project driven production planning and modeling, ProPlanT (PROduction PLANning Technology). They considered three types of agents: a Production Agent that models the behavior of the operating units in a factory, a Production Management Agent that models the managerial level, and Project Planning Agent that carries out project configuration and design [15]. ProPlanT, unlike KARMEN, is more dedicated to model production units' behavior than monitoring process conditions that are abnormal. The DESIRE (DEsign and Specification of Interacting REasoning Components) system was designed for knowledge-intensive multi-agent systems to model distributed industrial and business processes. Its main component is the Control System Interface Agent that monitors and analyzes network components to determine whether there may be a fault. This agent activates two diagnosis agents that determine the cause of the problem. A service restoration agent generates a plan to restore the network once the fault has been determined [5]. The MAGIC (Multi-Agents-based Diagnostic Data Acquisition and Management in Complex System) system also provides a real time agent architecture to diagnose abnormal and faulty conditions in industrial processes [14]. DESIRE and MAGIC are similar to KARMEN in monitoring components to detect faulty conditions. However, they differ in the fact that KARMEN supports notification of designated operators when an event occurs. Also, expert system technology has been applied in the process control industry to address abnormal situation management issues including fault detection, diagnosis, and alarm screening [10, 11]. Expert systems and the other technologies mentioned above include diagnosis of abnormal conditions which KARMEN currently lacks. We are planning to investigate this issue in the near future. The advantage of KARMEN over the others is the support for process operators to be appropriate notified when an event of concern happens.

6. SUMMARY

The KARMEN multi-agent system for process monitoring and notification introduces the concept of combining dynamically deployed condition monitoring with policy driven notification. The result is live remote monitoring capabilities available to all plant personnel, enabling users to register an interest in specific aggregate process conditions and be notified appropriately as the conditions change. Such condition monitoring has promising applications in early detection of abnormal conditions that can compromise plant safety and profitability. Policy-directed notification allows for representation of both organizational requirements and personal preferences concerning the mode and salience of user messaging.

In the future, we will enhance KARMEN by allowing plant operators to specify complex monitoring conditions using an intuitive graphical diagramming tool. Also, we will include diagnosis of abnormal conditions in industrial processes.

7. ACKNOWLEDGMENTS

Research and development were conducted through participation in the Hydrogen Research at Florida Universities program sponsored by the National Aeronautics and Space Administration; NASA grant number NAG-3-2751. The University of West Florida and IHMC's participation in this program was made possible through contract UCF 26-56-208 with the University of Central Florida's Florida Solar Energy Center as a State University System research partner.

8. REFERENCES

[1] Aspen Technologies Hysys Dynamics. http://www.aspentech.com/.

[2] Blevins, T., McMillan, G., Wijsznis, W., and Brown, M. 2003. *Advanced Control Unleashed: Plant Performance Management for Optimum Benefit*. The Instrumentation, Systems, and Automation Society. 163-182.

[3] Bradshaw, J. M., Uszok, A., Jeffers, R., Suri, N., Hayes, P., Burstein, M. H., Acquisti, A., Benyo, B., Breedy, M. R., Carvalho, M., Diller, D., Johnson, M., Kulkarni, S., Lott, J., Sierhuis, M., & Van Hoof, R. 2003. Representation and Reasoning for DAML-based Policy and Domain Services in KAoS and Nomads. In *Proceedings of the Autonomous Agents and Multi-Agent Systems Conference*. Melbourne, Australia. ACM Press, New York, NY, 835-842.

[4] Bradshaw, J. M., Beautement, P., Breedy, M., Bunch, L., Drakunov, S. V., Feltovich, P. J., Hoffman, R. R., Jeffers, R., Johnson, M., Kulkarni, S., Lott, J., Raj, A., Suri, N., & Uszok, A. 2003. Making Agents Acceptable to People. In *Intelligent Technologies for Information Analysis: Advances in Agents, Data Mining, and Statistical Learning*, N. Zhong and J. Liu, Eds. Springer Verlag, Berlin, in press.

[5] Brazier, F., Dunin-Keplicz, B., Jennings, N., Treur, J. 1996. Modeling Distributed Industrial Processes in a Multi-Agent Framework. In *Towards the Intelligent Organisation: a Coordination Perspective*, G. O'Hare and S. Kirns, Eds. Springer Verlag.

[6] Carvalho, M. and Breedy, M. 2002. Supporting Flexible Data Feeds in Dynamic Sensor Grids Through Mobile Agents. In *Proceedings of the 6th International Conference in Mobile Agents*, Barcelona, Spain, October 2002.

[7] Damianou, N., Dulay, N., Lupu, E., and Sloman, M. 2000. Ponder: A Language for Specifying Security and Management Policies for Distributed Systems. Imperial College, UK, Research Report Department of Computing 2001.

[8] Emerson Process Management DeltaV Distributed Control System. http://www.easydeltav.com/.

[9] Glymour, C., and McGlaughlin, K. 2003. Analyzing A Data Lookup Method for Machine Learning in Monitoring and Fault Localization for Hydrogen Generation Plants, Chemical Processing Plants and Other Complex Systems. IHMC Final Report for UCF contract 26-56-208.

[10] Hamdy, N., and Fulvio, R. 2003. Abnormal Condition Management with Real-time Expert System and Object Technology. *PCAI*, 17(1), 28-35.

[11] Hansen, J., Bradshaw, J., Suri, N., Bunch, L., Pechoucek, M, Glymour, C., McGlaughlin, K., and Breedy, M. 2003. Software Agents and Knowledge Discovery and Data Mining for Enhanced Safety and Control of Hydrogen Operations. NASA Hydrogen Research at Florida Universities, Technical Report for grant NAG-3-2751 (August, 2003).

[12] Jennings, N., and Bussmann. S. 2003. Agent-Based Control Systems. IEEE Control Systems Magazine 23(3).

[13] Johnson, M., Chang, P., Jeffers, R., Bradshaw, J., Soo, V., Breedy, M., Bunch, L., Kulkarni, S., Lott, J., Suri, N., and Uszok, A. KAoS Semantic Policy and Domain Services: An Application of DAML to Web-Services-based Grid Architectures. In *Proceedings of the AAMAS 03 Workshop on Web Services and Agent-Based Engineering*, Melbourne, Australia, July 2003.

[14] Köppen-Seliger, B., Ding, S., and Frank, P. 2001. EU IST Programme: Proposal Submission and Two Successful IAR Initiatives "MAGIC" and "IFATIS". Plenary lecture IAR annual meeting, Strasbourg.

[15] Marík, V., Pechoucek, M., Štěpánková, O., and Lažanský, J. 2000. ProPlanT: multi-agent system for production planning. *Applied Artificial Intelligence Journal*, 14(7), 727–762.

[16] OLE for Process Control. http://www.opcfoundation.org/.

[17] Schreckenghost, D., Martin, C., and Thronesbery, C. 2002. Specifying Organizational Policies and Individual Preferences for Human-Software Interaction. In Etiquette for Human-Computer Work, Papers from the AAAI Fall Symposium. Technical Report FS-02-02, AAAI Press.

[18] Suri, N., Bradshaw, J., Breedy, M., Groth, P., Hill, G., and Jeffers, R. 2000. Strong Mobility and Fine-Grained Resource Control in NOMADS. In *Proceedings of the 2nd International Symposium on Agents Systems and Applications and the 4th International Symposium on Mobile Agents*. Springer-Verlag.

[19] Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., Hayes, P., Breedy, M., Bunch, L., Johnson, M., Kulkarni, S., and Lott, J. 2003. KAoS policy and Domain Services: Toward a Description-logic Approach to Policy Representation, Deconfliction, and Enforcement. In *Proceedings of IEEE Fourth International Workshop on Policy*. Lake Como, Italy, June 2003, IEEE Computer Society, Los Alamitos, CA, 93-98.