

ALARM ROOT CAUSE DETECTION SYSTEM

Milan Rollo, Petr Novák, Jiří Kubalík, Michal Pěchouček
*Gerstner Laboratory,
Department of Cybernetics, Faculty of Electrical Engineering,
Czech Technical University in Prague,
Technická 2, 166 27, Prague 6, Czech Republic
{rollo|novakpe|kubalik|pechouc}@labe.felk.cvut.cz*

Production process control becomes complicated as the complexity of the controlled process grows. To simplify the operator's role many computer based control systems with integrated visualization clients have been developed. In many practical circumstances a malfunction of one or more process components results in other related components entering alarm states. Several alarms appear on the control operator's display in a short time making it difficult for the operator to diagnose the root cause quickly. Within this paper we describe a solution of this problem based on a multi-agent system that processes all incoming alarms, identifies the root cause alarms vs. alarms arising in consequence of the roots and presents the diagnostic results to the operator visually.

1. INTRODUCTION

Nowadays, as the complexity of the controlled processes grows, more stress is laid down on the operator. Number of manufacturers offers software for automatic process control with built-in visualization clients or even complex solutions including controllers, sensors, communication buses and visualization devices (e.g. control panels, touch screens). All these components are dedicated to support the operator's role. They allow the operator to select the controlled component and assign to it limit values of variables. When the variable exceeds the selected value (or comes down under it) the operator is informed about it visually. This helps him to quickly recognize the origin of the problems and fix it up.

But the situation becomes more complicated for the operator when malfunction of single process component results in other related components entering the alarm states. In such a case several alarms may appear on operator's display simultaneously or in a short time and in a random order (which depends on the nature of the variables and speed of the sensors). Some of the alarms may not appear on the screen at all, because the control process is displayed schematically only and contains just principal components (due to the clarity reasons). This all makes it

difficult for the operator to recognize the origin of the problem. In this paper we describe a diagnostic system that supports the operator's decision making when multiple components raise alarms simultaneously. Main reason to develop this system was, that in some kinds of production processes is necessary to fix up the problems as soon as possible to avoid eventual exposure to danger or economical losses.

Solution is based on the multi-agent system that models the production process, processes all incoming alarms and determines the root cause. New determination algorithms, based either on the topology of the production process or physical nature of individual components, were developed to this purpose. Results of the determination process may be presented visually to the several operators. This diagnostic system nicely illustrates capabilities of the multi-agent system based solutions like a modularity or adaptability.

1.1 State of the Art

Use of multi-agent systems for modeling and controlling the production processes grows rapidly and seems to be very promising. In general the agents technologies are suitable for domains that possess either of the following properties: (i) where highly complex problems need to be solved or highly complex systems to be controlled or (ii) solving problems or controlling systems, where the information is distributed and is not available centrally. In manufacturing agent technologies have been applied mainly in planning highly complex production, control of dynamic, unpredictable and unstable processes, diagnostics, repair, reconfiguration and replanning. Important application domains of agent-based applications can be also found in the field of virtual enterprises (e.g. forming business alliances, forming long-term/short-term deals, managing supply chains) and logistics (e.g. transportation and material handling, optimal planning and scheduling, especially in cargo transportation, public transport but also peace-keeping missions, military maneuvers, etc.)

There are several companies that have adopted the agent-based solutions in production already, e.g. Daimler-Chrysler car manufacturing (McFarlane, 2000), Rockwell Automation developed agent based solution for BHP Melbourne, Australia, or ExPlanTech/ExtraPLANT agent-based production system running in Modelarna Liaz pattern factory, CZ and Hatzapoulos, packaging company, Greece (Pěchouček, 2002). The most relevant is the application of agent technology in a Reconfigurable Shipboard automation system developed by Rockwell automation and Rockwell Scientific Company (Maturana, 2003). In this application the agents are brought down to the level of physical components of the shipboard chilling system and they monitor and control stability of the whole of the chilling machinery in distributed manner. The practical applications are in the focus of attention of the HMS consortium (Brennan et al., 2003)

2. SYSTEM DESCRIPTION

Infrastructure of the proposed diagnostic system is shown schematically in Figure 1. Within this infrastructure, a software agent models each process component. The agent relationships model the topology of the process component input/output

relationships. Each agent receives alarms activated by the component it represents and alarming agents collaborate with related agents to identify potential root causes.

Root cause detection algorithms utilize component topology, component type, and alarm type, with the goal of effectively diagnosing root causes without necessity of encoding expert knowledge specific to the particular process. Expert system based solutions require a separate knowledge base describing the relations between the components for each process control case. Such an expert system may not be available for the particular process or may contain incomplete information. Once such a system is build up for a concrete process it provides a operator with most reliable results. However it doesn't cover the cases that occurred in the process never before. In contrast, the proposed agent based solution is general enough and is expected to work for an arbitrary manufacturing case, provided that there are agents that represent all of the components available.

The agent-based algorithms developed to date include rule-based and topology-based search methods, which appear to have promise for this purpose (see section 4).

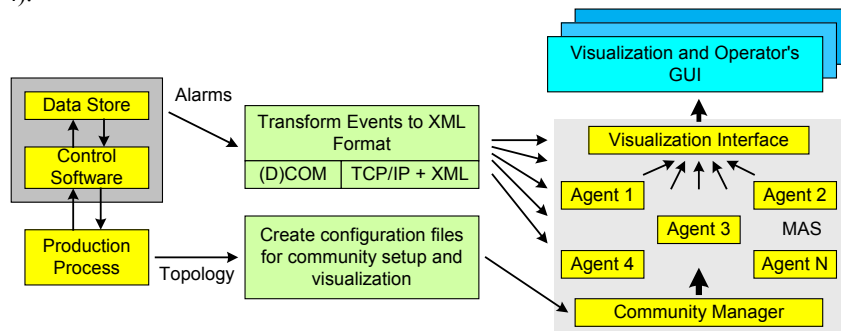


Figure 1 - Configuration of the multi-agent alarm root cause diagnosis system

2.1 Production Process Interface

In the past the individual production processes were isolated and controlled by software that used different forms of communication and data management. Technical progress in the area, especially the distributed control, brought couple of problems with interconnection among different production processes and control software. Industry leading companies thus formed a foundation that is dedicated to develop standard specifications. Most widely used solution and de facto standard in this area is currently the OPC (OLE for Process Control) (OPC, 2004).

OPC is based on the OLE/COM standard designed by Microsoft. The OPC Alarms and Events specification is intended for use with process automation systems that generate alarms and events. An OPC Alarms and Events server is written to expose the alarms and events present in the system. It does not create the alarms and events; it only reports the alarms and events previously defined in the system using a standard communication interface. Once defined, the alarms and events in the system are automatically generated, based on the operating conditions and actions performed in the process plant. The OPC Alarms and Events server captures these alarms and events automatically, and makes them available to any client application that is interested in this information.

OPC in this case serves as a kind of database (Data Store) that all clients can utilize. Information is in this database filled by the control software that gets it from the sensors (e.g. through the communication bus).

In this project we use the OPC Alarms and Events server to receive alarms invoked in the process (either real process or its simulation). These alarms (each containing information about its source, type, date of rise, variable, etc) are transformed to the XML format and send to the multi-agent system to be diagnosed.

2.2 Multi-Agent System

Alarm root cause determination process itself is carried out by the multi-agent system. Input of the determination process, provided to the system by the OPC-XML interface, is alarm record from OPC server converted to the XML format.

Using multi-agent system brings us a couple of advantages in comparison with using an expert system only:

- **Prediction ability** - system can predict the problems (determine root causes) in process only from the general rules or process topology and thus discover problems that appeared never before and will not be covered by the expert system rules.
- **Adaptability** - system can be easily fit on different process (unlike expert system that is tied to the concrete process).
- **Modularity** - process can be simply extended with a new type of the component - adding some new rules based on the physical characteristics of this component we can predict the behavior of the process (even when we have no knowledge about the entire process). Using the expert system user is enforced to add a concrete rules based on the observation of the process's behavior.

2.3 Operator's Visualization Interface

The result of root cause detection algorithm is represented by output state of one or more agents. This should be displayed on the operator's screen in a comprehensive way. When a complex system is to be controlled a number of output windows providing different views and details of the process may be required. For this purpose a versatile visualization system (VISIO) has been developed. It enables among others to define multiple snapshots of different parts of the process; they can be displayed to the operator either all at the same time or just some of them.

Developed visualization system contains also multiple-operator support. Each operator can be informed only about events that happened in the part of the production process he is responsible for.

3. MULTI-AGENT SYSTEM IMPLEMENTATION

Agent community consists of four different types of agents (see Figure 2). Besides the three static agents (Agent Factory, Root Cause Analysis Agent and Visualization

Agents), each of them appears in the system only once, there are several Block Agents. Their number depends on the particular production process.

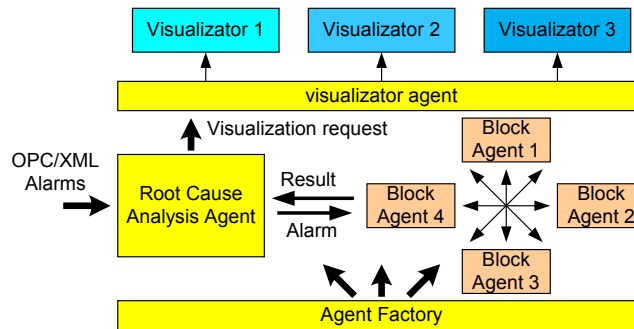


Figure 2 - Design of the multi-agent system

3.1 Graph Representation

The process is represented by agents connected into an oriented graph. Every component (pump, valve, tank, controller, etc.) is represented by one agent (Block Agent). Agents are connected by oriented links representing input/output relationships among components.

Community manager (Agent Factory) generates agents according to a configuration file. This file contains a record for each component with its name, agents linked to its input/output, info whether it generates alarms or not, type of component it represents (sensor or other), the variable it monitors, etc. Note that the links connecting agents represent both material flow as well as control links present in the process (e.g. feed back control signals). This is important to make sure that no causal relation among components would be omitted when looking for the root cause alarm.

3.2 Agent Types

- **Agent Factory** - community manager that is used to startup the agent's community. This agent starts up the agent platform providing necessary infrastructure, Root Cause Analysis Agent, Visualization Agent and certain number of Block Agents.
- **Root Cause Analysis Agent (RCAA)** - coordinator of the alarm determination process. It is connected to the OPC-XML interface and receives all alarms that arose in the process from the OPC Alarms and Events server. Alarms are stored in the internal queue and processed on the FIFO basis. Each alarm is checked against the rules in the internal expert system first. In case that no matching rule is found, appropriate agent (corresponding to the source of alarm in the process) is informed about the event. Agents process the alarm using one of the inference modes and RCAA receives the result. Request to display this result on the operators' screens is with that send to the Visualization Agent.
- **Visualization Agent** - is responsible for displaying the information about the alarm states of all function blocks on screens of all operators that are involved

in. Agent connects to the VISIO program (see section 2.3) that is running on the operator's side. Agent gets information about these connections from the Agent Factory (it is stored in the visualization configuration file). When new information about change of function block's alarm state arrives from the RCAA, Visualization Agent transforms it into format acceptable by the VISIO and sends it to all operators that has a particular function block on their screen.

- **Block Agent** - every Block Agent represents a real physical entity in the process. Block Agent has a record about its inputs, outputs and its alarm state. It has also assigned functionality in the process (e.g. pump, valve, tank, etc.). When agent receives information about new alarm it invokes an inference process to determine the alarm type.

4. INFERENCE MODES

Multi-agent system supports two inference modes: topology-based search and rule-based search. Topology-based search is based only on the topology of the process, rule-based exploits some background (prior) knowledge about the process. This knowledge is stored in form of rules in the rules database.

4.1 Topology-Based Search

Topology-based search takes advantage of converting the process model into its graph representation. This algorithm is based on the input/output relationship only and doesn't take into account features of the particular process. This algorithm is divided into two separate branches:

Backward (Upstream) Search:

- Starting from the currently processed function block, it goes back through all its inputs (as long as the block has some).
- Looks for the possible cause of the alarms.
- Algorithm runs until the first block with alarm in the chain is found (this block will be marked as root).

Forward (Downstream) Search:

- Informs all successors (all agents in the output direction) that they are not root alarms any more.
- Updates alarm types for visualization.

Finally results of the both search branches are merged together. In this result each agent (function block) has its own record that contains its name, type of the determined alarm type and information whether the agent can generate alarm. These records are stored in the tree form.

The alarms are stored in a buffer as they come and are processed sequentially one by the other. The final result of the algorithm does not depend on the order in which the alarms are processed.

4.2 Rule-Based Search

Obviously, the topology as such captures only the lowest-level information of the monitored process. This makes the root cause detection algorithm process independent. On the other hand, the topology itself cannot provide sufficient bases for correct root cause detection when complex processes are given. It turns out that some form of rules, which take into account knowledge of the process may help to make the root cause detection more reliable.

A rule-based search utilizes agents that are specialized to model specific component types (e.g. pump, valve, tank, heat exchanger). The agents apply first principle rules based upon the type of alarming component, the alarm type, and the component topology. For example, in the case of high pressure/level alarms from a tank a search is performed for other alarming components contributing to high mass/energy input and low mass/energy output.

Each agent has assigned a type of the component it represents (sensor, pump, valve, etc.). Only agents that represent sensors can signal alarm. Each sensor has assigned a variable it monitors (pressure, temperature, etc.). When a new alarm arises, rule-based search is invoked first. This means that the corresponding agent searches its rule database for any rule whose condition part can be satisfied. If the agent finds such a rule, it carries out actions specified in the action part of the rule. In case the agent has not got any rule that could be applied at the given moment the topology-based root cause detection algorithm is launched. Thus the topology-based search is considered to be a complementary option to the rule-based one. It is invoked only when the rule-based search fails to run.

5. EXPERIMENTS

The proposed diagnostic system was originally developed to improve the control process in hydrogen production plant. Because this process is very complex, simplified case study based on the distillation column was chosen to execute the experiments (both processes have similar physical characteristics).

Instead of connection to the real physical process, mathematical model of the distillation column running in the Hysys (Hysys, 2004) simulation environment was used. This allowed us to better simulate the malfunctions and alarm explosions in the process. As control software the DeltaV Automation System (DeltaV, 2004) was used. Simulation environment and control software were connected via the OPC Server. Diagnostic system was also connected to this server in order to receive the alarm records.

Distillation column used in this case study consists from more than fifty components (function blocks). We also carried out some scalability tests (measuring the number of messages and operational memory consumption), with several hundreds of components.

Multi-agent system was implemented in Java using the JADE multi-agent platform (Jade, 2004). Other parts (Production Process Interface and Operator's Visualization Interface) were implemented in C/C++.

6. CONCLUSIONS AND FUTURE WORK

System described in this paper illustrates the capabilities of the multi-agent systems to solve problems in highly complex distributed environment. One of the features of the multi-agents systems we utilize in this case is modularity. It allows us to connect and disconnect new components and even the entire parts of the process dynamically.

Obviously, root cause alarm diagnostics based purely on the topology knowledge about the process cannot reliably return the optimal solutions due to the high complexity of physical systems. As the preliminary experiments showed the utilization of rule based search algorithm might considerably improve the performance of such a system.

There are still some open issues that need to be solved to make the diagnostic system more robust and reliable:

- Synchronization and cooperation between both inference modes (topology-based and rule-based search).
- Improve the connection with OPC Alarms & Events server - suppression of alarms identified as consequence of the root alarm in the DeltaV.
- Increase of alarm priorities (operator's notification level) when alarm isn't suppressed within a certain period of time (e.g. when high pressure alarm on the column will arise, alarm priority will be increased every hour until the alarm will be suppressed).

7. ACKNOWLEDGEMENT

The project work has been in part co-funded by NASA Hydrogen Research Effort - Software Agents and Knowledge Discovery and Data Mining Research for Complex System Safety, Health, and Process Monitoring project and by the Grant No. LN00B096 of the Ministry of Education, Youth and Sports of the Czech Republic.

8. REFERENCES

1. Brennan R., Hall K., Mařík V., Maturana F., and Norrie D. A real-time interface for holonic control devices. In Mařík, McFarlane, and Valckenaers, editors, *Holonic and Multi-Agent Systems for Manufacturing*, number 2744 in LNAI, pages 25–34. Springer-Verlag, Heidelberg, June 2003.
2. DeltaV - the Digital Automation System for Process Control. <http://www.easydeltav.com>, 2004.
3. HYSYS - Integrated Simulation Environment. <http://www.hyprotech.com/hysys>, 2004.
4. JADE - Java Agent Development Framework. <http://jade.tilab.com>, 2004.
5. Maturana F., Tichý P., Šlechta P., and Staron R. A highly distributed intelligent multiagent architecture for industrial automation. In Mařík, Muller, and Pěchouček, editors, *Multi-Agent Systems and Applications III*, number 2691 in LNAI, pages 522–532. Springer-Verlag, Heidelberg, June 2003.
6. McFarlane D. and Bussmann S. Developments in holonic production planning and control. *International Journal of Production Planning and Control*, 11(6):552–536, 2000.
7. OPC Foundation. <http://www.opcfoundation.org>, 2004.
8. Pěchouček M., Říha A., Vokřínek J., Mařík V., and Pražma V.. Explantech: applying multi-agent systems in production planning. *International Journal of Production Research*, 40(15):3681–3692, 2002.