Using Expert System and Object Technology for Abnormal Condition Management

Hamdy Noureldin and Fulvio Roveta
Gensym Corporation

BIAS 2002 International Conference
Milano, Italy
November 19-21, 2002

Table of Contents

Abstract

This paper discusses the problem of Abnormal Condition Management (ACM), defines the requirements for addressing this problem, and presents an application, developed using Gensym's Optegrity platform, which provides generic objects for managing abnormalities on heaters. The goal of this application is to sustain operational performance and maintain continuous availability by detecting and resolving abnormal process conditions early – *before* they impact operations. The heater models developed for the first application can be easily reused and adapted to other heating devices by customizing the objects with graphical tools.

The first application of these "generic heaters" has been installed in a refinery in the Middle East, and it is currently in the process of being deployed at other sites. A total of 80 preconfigured faults have been included for identifying the root cause of various heater problems. The application includes almost 240 messages that can be presented to operators for assisting with the diagnosis of problems and for providing guidance to quickly return to normal operation.

As part of the justification of this project, a return on investment analysis was completed. The payback period was estimated to be in the range of 3 to 8 months, depending on the type of heater, the application of the heater and the existing operating conditions.

Abnormal Condition Management

The more time it takes to discover and correct abnormal process conditions, the greater the loss and disruption to business operations. Abnormal conditions range from those that cause lower quality or reduced production rates to those that cause catastrophic shutdowns. These conditions can result from equipment failure or degradation, variability in raw materials, process drift, and operator error.

Abnormal conditions, also known in the process manufacturing industries as abnormal situations, are typically caused by a combination of events that are not normally expected to occur at the same time. These are not adequately addressed by the safety interlocks and exception logic in a conventional control system and may be difficult for the operators to detect and resolve.

Managing abnormal conditions is harder than ever. In the chemicals, oil, and gas industry, for example, the number of control loops that operators must manage has increased from two hundred to eight hundred per operator over the last twenty years. Increasingly complex processes and sophisticated control strategies contribute to the problem. It is not possible for operators to pay sufficiently close attention to every aspect of the operation.

The industry needs techniques to reliably warn of impending problems before they cause off-normal operation. Control systems are often limited in their ability to diagnose and correct process problems. They give the operator little help in determining why "*x is too high*". Limit excursions are symptoms, not root-cause problems. Alarms indicate that a problem exists, but they do not point out the source of the problem, nor do they explain the best course of action. Part of the problem is the number of standing alarms that operators must manage and the alarm floods that result when problems occur. Too much time is spent trying to figure out what went wrong. Even after the root cause is identified, operators do not necessarily execute the ideal corrective response. Delays and inappropriate responses are costly: minor problems can quickly escalate.

The consequences of abnormal process conditions are significant and they include:

- off-specification production
- waste
- expensive unplanned shutdowns
- schedule delays
- equipment damage
- environmental risks
- broken product schedules
- safety problems

Many operations managers can gain more from minimizing unplanned shutdowns and off-specification production than they can from applying advanced forms of process optimization. A single shutdown can wipe out all those hard-won gains. The impacts can be substantial – in the worst cases major accidents can lead to disastrous safety and environmental consequences. Preventable industrial accidents occur too often.

Existing Protection Layers

Process-manufacturing plants typically have multiple layers of protection against the costly and sometimes dangerous impacts of abnormal conditions. The first layer is the process control system, which is usually a Distributed Control System (DCS) or a Programmable Logic Controller (PLC) based system. It provides safety interlocks and exception logic. The next layer of safety may be provided by a dedicated "Safety Shutdown system," which shuts down a process when the control system is unable to do so in an emergency situation. Another layer of safety might be a fire and gas protection system. Not all of these protection layers are required for all processes.

All of these protection layers generally work in reactive mode and provide little proactive guidance to operators.

Most manufacturing sites place safe operations as high a priority as any. To help achieve safe levels of operations, manufacturers generally establish manufacturing practices that include automatic control. Secondly, they often establish safety departments, whose functions include promotion of safety and preparation for emergency conditions.

DCS- and PLC-based programmable control systems provide facilities for various alarm functions and shutdown routines to take care of many abnormal conditions. Control applications ensure operating the plants in a steady state in response to disturbances. They include regulatory control, advanced regulatory and multivariable constraint control. However, these systems usually do not provide help in:

1. Distinguishing between a false alarm, resulting from faulty primary elements, and a real alarm.
2. Filtering critical alarms from those that are less significant
3. Identifying the root cause of the alarm by interrogating all possible reasons for an alarm and selecting a particular cause for the alarm.
4. Detecting the root cause of the disturbances that shift from the normal operation.
5. Predict abnormal conditions before they are likely to happen.
6. Provide guidance to operators and safety personnel.

Commercially available software packages that offer abnormal condition management capabilities are now addressing the shortcomings of the conventional control systems.

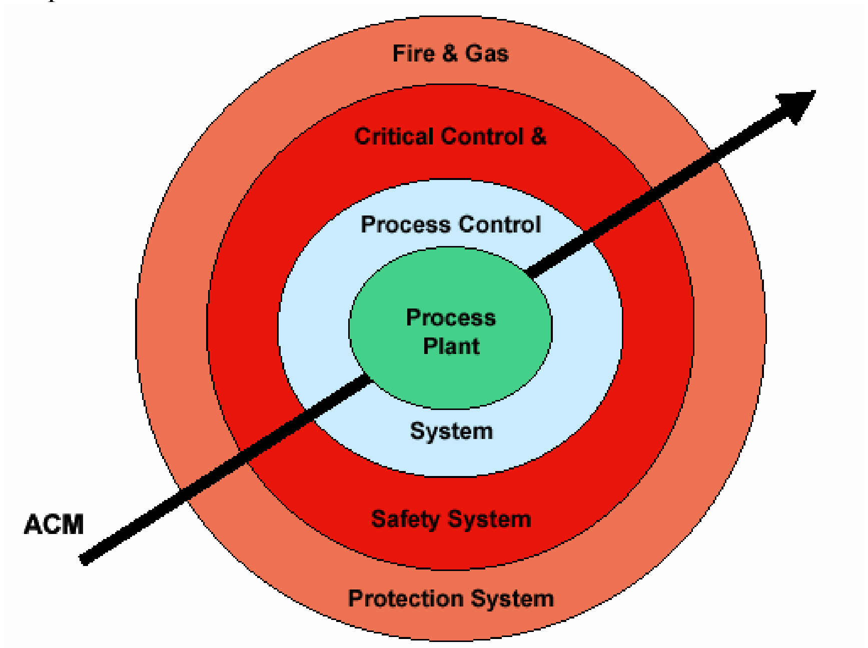
Proactively Managing Abnormal Conditions in Real Time

To go beyond the reactive mode of the traditional protection layers and provide operators with proactive assistance, what is needed is the ability to detect unusual events early, assess the potential impacts of those events, diagnose the root cause behind high-impact events, provide operators with advice, and if appropriate automatically take actions. Simply put, the logic flow is event detection, assessment, diagnosis, advice, and actions.

Expert systems enable software applications that provide proactive assistance to operators during abnormal conditions. Expert systems, which reason about data in a similar manner as human experts, have been a proven technology for over 15 years. Real-time expert systems add the dimension of time. Real-time expert systems reason about time-based data such as that generated through process control systems.

Real-time expert systems reason about data by capturing and applying expert knowledge through object-oriented models, rules, and procedures. Recent developments in expert system techniques are simplifying the ability to build and deploy applications for abnormal condition management in the process industries. These techniques are employing new graphical-language methods that allow experts to more quickly and easily represent the knowledge needed to provide proactive assistance in real time.

Abnormal condition management applications work in an anticipatory mode across the various safety protection layers within a manufacturing operation. In most applications, abnormal condition management acts in an advisory role to operators and safety personnel. However, there are situations where an abnormal condition management application could take corrective actions automatically, where little potential for hazardous conditions exists, such as set-point adjustments to improve quality or throughput. Abnormal condition management functionality does not replace any of the safety protection layers but complements them.



.

Abnormal Condition Management Automated Safety Protection (Courtesy of Arc Advisory Group)

Case Study: "Generic Heater" Objects for Abnormal Condition Management

Gensym Corporation has been implementing "smart" equipment objects that provide proactive diagnostic and management of abnormal conditions, and performance indices for a wide variety of applications that include safety, reliability, economics, operations, and process. This case study describes a generic and configurable smart object representing a heater. Such a "Generic Heater" is based on Gensym's Optegrity software and has been developed by Gensym's Professional Services Group using their experience in process, control, operations, maintenance, cost, and safety.

A smart object covers the following areas of functionality:

- Managing abnormal conditions
- Providing generic features
- Encapsulating smart knowledge
- Applying root cause analysis
- Utilizing technology and tools

The case study described here has been developed for a refinery, but the objects are designed for reuse with any type of a plant in which heaters are used. It is a generic application for all types of heaters in a wide spectrum of industries:

- Boilers
- Furnaces
- Ethylene Crackers
- Incinerators

The "Generic Heaters" can cover different designs:

- Cylindrical heaters
- Box heaters
- Multi-cell (up to 4 cells)
- Multi-pass (up to 8 passes)
- Heater with or without steam convection section
- Air pre-heater or forced and induced draft.

They can be used in different industries and applications:

- Ethylene, styrene in petrochemical plants
- Boilers in utility plant
- Crude CCR platformers
- Hydrocrackers in conversion refinery
- FCC's, etc.

Generic heaters are software objects containing diagnostic, fault models and advisory messages for managing over 80 faults typical for such devices. They provide built-in and external calculations able to generate three different classes of messages on the operator screens:

- Diagnostic
- Root cause analysis and identification
- Recovery actions.

These generic heaters are out-of-the-box objects, immediately usable. They integrate the interface to the field, smart sensors, generic event generation, process knowledge, root-cause analysis, message generation and GUI (Graphical User Interface).

The objects are graphically developed, with no rule or procedural programming. Any fault model, check and message generation can be configured just by point and click.

This application has a very short payback period. Depending on the plant, payback is expected to be within three to twelve months.

Project Execution

Developing an expert system application involves a different development methodology compared to projects based on traditional software. Expert systems often rely on information that is unknown or not formalized before the project begins. One of the most critical tasks is the knowledge elicitation, or capturing a human expert's knowledge and translating it into expert system rules, procedures, and object-oriented models. Application development is usually managed incrementally – first starting from a small kernel of basic features and then rapidly testing and adding new capabilities. Writing the software is not the central task in the application development – it is part of a process that aims to make the knowledge visible.

The primary steps in the "Generic Heater" application development included the following:

- Knowledge elicitation
- Fault models implementation and testing
- Operator interface design and development
- Field interface development

Knowledge Elicitation

Knowledge elicitation was a fundamental phase in the project. Using all possible knowledge sources (operations manuals, specifications, and human experts) a complete description of the necessary knowledge was extracted for use with the application development.

The knowledge elicitation process was organized according to the following step-by-step procedure:

1. Acquiring a general understanding of the domain: this was a critical phase in the process, where the scope of the application and the overall process operations were defined. This was achieved by finding answers to the following simple questions:

    - What does the system do?
    - How does it do it?
    - How is it managed now?
    - What are the major sub-systems and how do they interface with each other?

2. Identifying the sources of domain knowledge: collecting domain knowledge required the identification of any possible information source. Since the project is expertise-based, this means that technical documents, such as user guides, maintenance and troubleshooting manuals, were integrated with human expertise. For such a reason Gensym worked jointly with the end user's experts (refinery and process control department personnel) and, for some specific situations, also with heater's manufacturer experts. The team included operational experts, process engineers, heater specialists, control and application engineers and IT specialists.

3. Identifying general problems: the development team used their experience and the client organization's resources to identify and characterize important problems occurring in the heaters, how these problems are dealt with in the present set up, and what could be done to improve the fault management process. This was achieved by addressing the following issues:

    - Highlighting the most frequent problems (faults) in the domain

- Highlighting the most important problems in the domain, i.e., the ones with the most significant impact
- Running through several specific scenarios of problems documenting when they occur and how they are manifested (i.e., their symptoms)
- How are these problems detected now? What is good and bad about the present procedure?
- How can we distinguish the real causes of the symptoms (i.e., root causes) from other problems?
- What corrective actions can be taken?
- Who are the targeted users for the application? What kind of reports do they expect to be generated?

4. Describing the domain: The domain model describes the system configuration and relations among specific components. For this application the domain model must describe the most generic system configuration and relations in the set of manageable heaters. Developing this model required acquiring the following information:

   - Which specific equipment can be used in a heater?
   - What is the system configuration (i.e., component interconnections, logical topology)?
   - How can data be accessed?
   - Which specific relationships exist among components?

   The "generic" heater has been split in several subsystems. Every subsystem has a set of parameters that are only related to this system. They are combined together for ease of configuration and for selecting the type of heater according to the one existing in the plant. These subsystems include, but are not limited to:

   - Steam System. This includes the economizer, lower steam generation coil, upper steam generation coil, and super-heater.
   - Air System. This includes airflow to the heater, air temperature from air pre-heater, air registers, forced draft.
   - Fuel Gas. This includes fuel flow, temperature, specific gravity, burner system.
   - Fuel Oil. This includes flue gas temperature, flue gas composition, temperature, stack damper, and induced draft.
   - Process. This includes charge inlet temperature, charge flow, charge outlet temperature.
   - Stack and Flue Gas. This includes flue gas temperature, flue gas composition, temperature, stack damper, induced draft.
   - Control. This includes control loop modes, and output value, set points outlet temperature loop, fuel pressure or flow loop, process fluid flow loop.

   Each subsystem includes all the possible information actually available, and can be configured independently from the others. Moreover the radiant section was split into 4 cells, and up to 8 passes can be configured, in order to make the application as flexible as possible.

5. Defining the indices for Performance and Abnormalities Management. The Generic Smart Heater has both the scope of managing abnormal conditions and optimizing

performance. The definition of indices for identifying lack of performance, or abnormal conditions, and the actions to be taken in response to out-of-range indices required additional study.

When an abnormal condition is identified, the system must react, according to the problem type, to reach the following states:

- *Bring the system to a safe state without requesting a plant shutdown*. The smart heater object can proactively identify situations that can lead to an unplanned shutdown, for instance as a result of increased outlet temperature (high skin temperature, high box pressure, low charge flow, and minimum fuel pressure, etc.), and then assist in recovering in order to avoid the shutdown.
- *Avoid and/or minimize the impact of disasters, accidents, leakage, fire, explosion, and backfiring*. Actions were defined to manage critical and potentially dangerous situations that are triggered through out-of-range indices. Some examples are: adjusting heater operation for an out-of-range fire/tube leakage index; adjusting position of air and stack damper for out-of-range explosion index and low O_2 %; adjusting burner operation for out-of-range burner pressure and flame index, fuel gas pressure, or fuel leakage.
- *Respect all environmental indices to avoid violation of emissions.* Violations of environmental indices, such as the emission index or heat release index, will lead to the Optegrity application adjusting heater and burner operation.
- *Return to normal and keep operation in the normal envelope minimizing the violations.* Different types of abnormalities are considered (reliability, economic, operation, process, etc.) with suggested mitigation procedures.

Another very important aspect of operations where the Optegrity smart heater application adds valuable capabilities is plant performance. Obtaining compliance to performance indices will bring economic return due to a more efficient heater utilization and less unplanned activity interruption. The application has been designed to support different types of compliance:

- *Reliability indices* to reduce equipment damages, deficiencies, failures, and shutdown.
- *Economic performance indices* to maximize the return on the investment (maximize revenues minimizing costs)
- *Operation indices* to work in the operating envelope that ensures effectiveness of operation with maximum up time.
- *Process indices* assigned by planning and process engineering. (Key Performances Indices and Critical Performance Indices.)

The complete application analysis and knowledge elicitation required about 8 man weeks, split in 4 weeks for the project analysis and 4 weeks for the knowledge elicitation.

Fault Models implementation and testing

The fault models implementation activity has been simplified by the tools used in the application development. The platform used for software development and deployment is

Gensym’s Optegrity, which is based on the Gensym's G2’s expert system software. Optegrity allows rapid development of abnormal condition management applications through a graphical approach. Developing and testing the complete application took about 10 developer weeks.

Generic fault models were quickly developed using a graphical language with Optegrity called SymCure. SymCure provides a methodology and a framework for real-time fault management in large-scale systems. It addresses the full life cycle of problem identification based on *symptoms*, *diagnostic testing*, and *fault isolation*, through recovery. SymCure also protects the operator from “alarm flooding”. It is based on Generic Fault Propagation Models (GFPM), tied to an object-oriented domain representation and scalable algorithms. SymCure combines the generality of FMEA (Failure Mode and Effect Analysis) models with online, asynchronous event correlation and diagnosis.

A fault model in SymCure describes the propagation of failures via potential *failure paths* in the system. The paths model how fault events will cause other symptom events and test-result events. It is a directed graph (digraph) model, with the nodes representing fault, symptom, and test events, and the connections between the nodes representing the “cause and effect” relationships, or dependencies, among these events. The events represented in the model can take the values, “true”, “false” or “unknown”.

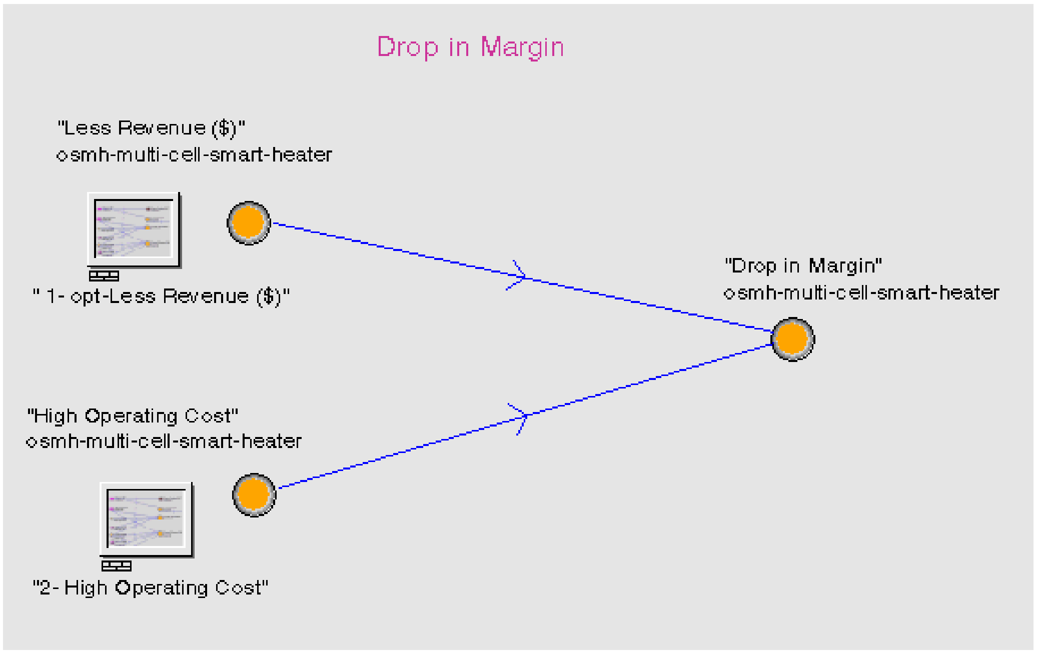
A *“Fault”* is an underlying independent root-cause problem. Faults have associated corrective or mitigation actions specified by procedures that can be started against a specific domain object. Actions can be defined to execute when a fault is suspected as a cause of the observed symptoms and when a fault is confirmed. For a fault event, a true value indicates belief that the fault has occurred and a false value indicates belief that the fault did not occur. A fault may be “suspect” if it is a possible cause of observed symptoms.

A *“Symptom”* is an effect of underlying faults in the monitored system. A measured symptom arrives at SymCure, unsolicited asynchronously from external systems. An unmeasured symptom status is used for failed sensors, and convenience in modeling, to represent those effects that are not measurable, but are important for fault propagation. For a symptom event, a true value means that the symptom has occurred and a false value means that the symptom did not occur within the time delays specified in the node connections.

A *“Test”* is an observable effect of faults in the monitored system, like a measured symptom. But, unlike symptoms, the observation can be requested at any time. Test results arrive asynchronously (and possibly unsolicited), just like a symptom. Tests have an associated set of actions applied to the monitored system, specified as an external procedure name that can be started against a specific domain object. Upon request, after some indeterminate amount of time, the result of the test is returned to SymCure as a truth-value. Test procedures may be fully automated, may simply be a request to an operator, or a combination of the two. The true or false value for a test indicates whether the test passes or fails, respectively. The notion of "test" is powerful and general, (Simpson and Sheppard, 1994) and can imbed arbitrarily complex analysis and actions, as long as it returns a single truth-value.

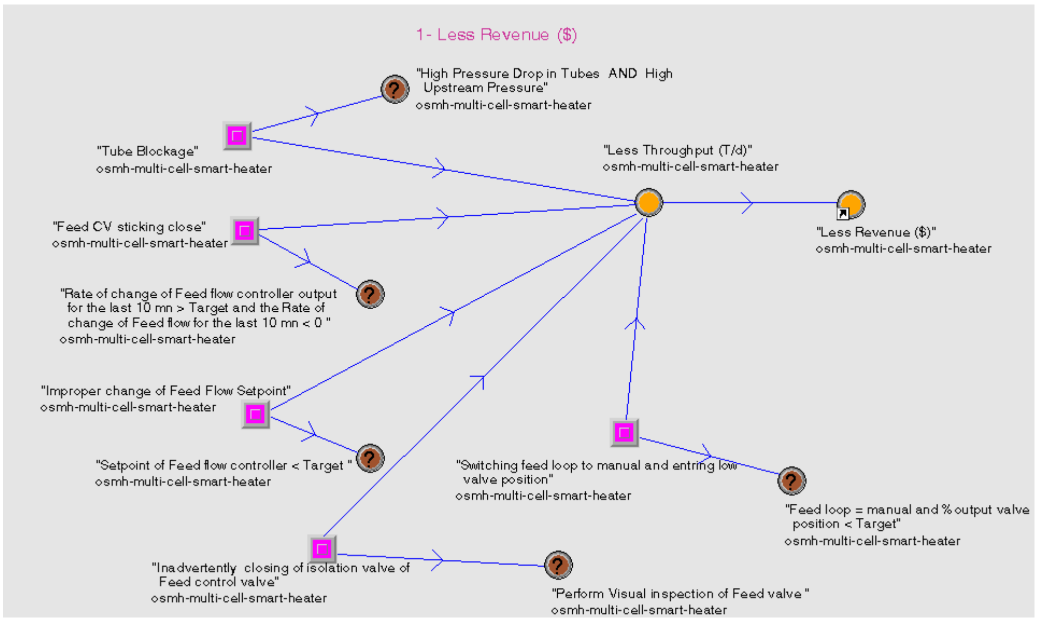
Since fault models are very simple to describe using this methodology, they were directly developed using SymCure's tools. Faults are organized in a hierarchical structure, starting from the most generic symptoms, and digging down to describe them in more details.
The fault models are defined at a generic level, that means they are applicable at any object belonging to the specified class (in this case the multi-cell smart heater). This implies that they are reusable for any object of the same class that is configured in the application.

As an example we may consider a very high level symptom: “Drop in Margin”, which means that economic targets are not met.


Drop in Margin
"Less Revenue ($)"
osmh-multi-cell-smart-heater
" 1- opt-Less Revenue ($)"
"Drop in Margin"
osmh-multi-cell-smart-heater
"High Operating Cost"
osmh-multi-cell-smart-heater
"2- High Operating Cost"

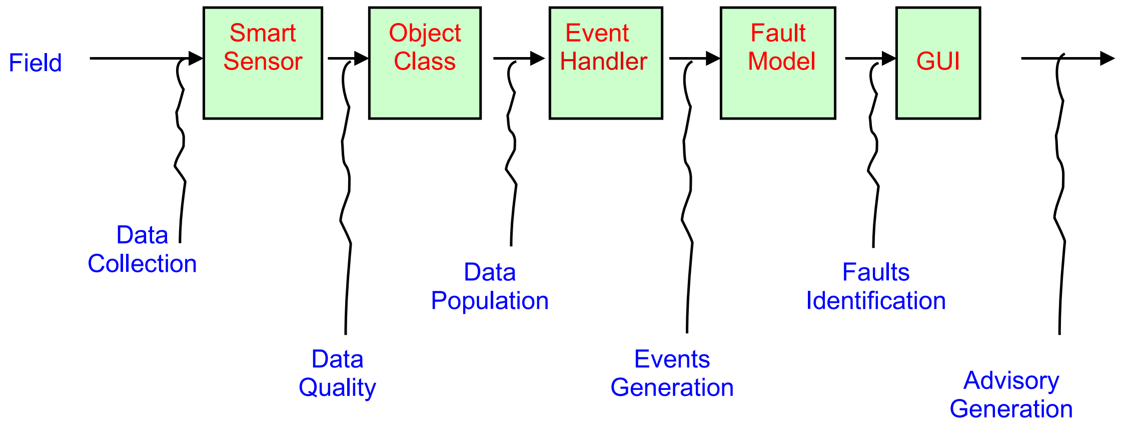
High-level generic propagation fault model for the “Drop in Margin” symptom

This symptom can be caused by other symptoms: “Less Revenue” or “High Operating Costs”. For each of the two symptoms fault models has been built to describe relationships with other symptoms or faults, until the root cause of the problem has been found


1- Less Revenue ($)
"High Pressure Drop in Tubes AND High Upstream Pressure" osmh-multi-cell-smart-heater
"Tube Blockage" osmh-multi-cell-smart-heater
"Less Throughput (T/d)" osmh-multi-cell-smart-heater
"Feed CV sticking close" osmh-multi-cell-smart-heater
"Less Revenue ($)" osmh-multi-cell-smart-heater
"Rate of change of Feed flow controller output for the last 10 mn > Target and the Rate of change of Feed flow for the last 10 mn < 0" osmh-multi-cell-smart-heater
"Improper change of Feed Flow Setpoint" osmh-multi-cell-smart-heater
"Setpoint of Feed flow controller < Target" osmh-multi-cell-smart-heater
"Switching feed loop to manual and entring low valve position" osmh-multi-cell-smart-heater
"Feed loop = manual and % output valve position < Target" osmh-multi-cell-smart-heater
"Inadvertently closing of isolation valve of Feed control valve" osmh-multi-cell-smart-heater
"Perform Visual inspection of Feed valve" osmh-multi-cell-smart-heater

Low-level generic propagation fault model, describing the "Less Revenue" symptom, possible faults causing the symptom, and related tests

Symptoms and tests are generated using "generic" block languages (i.e. not referred to specific object instances or tags, but to object classes and attributes), performing all the necessary data processing. The picture represents the logical sequence of actions performed by the system



Information handling in the application

Smart sensors preprocess values received from the field for data validation and reconstruction. Validated data are sent to the smart heater object. The Event Handler monitors and analyzes real-time data and trends to identify abnormal conditions and generating appropriate symptom events, which are sent as input to the fault models.

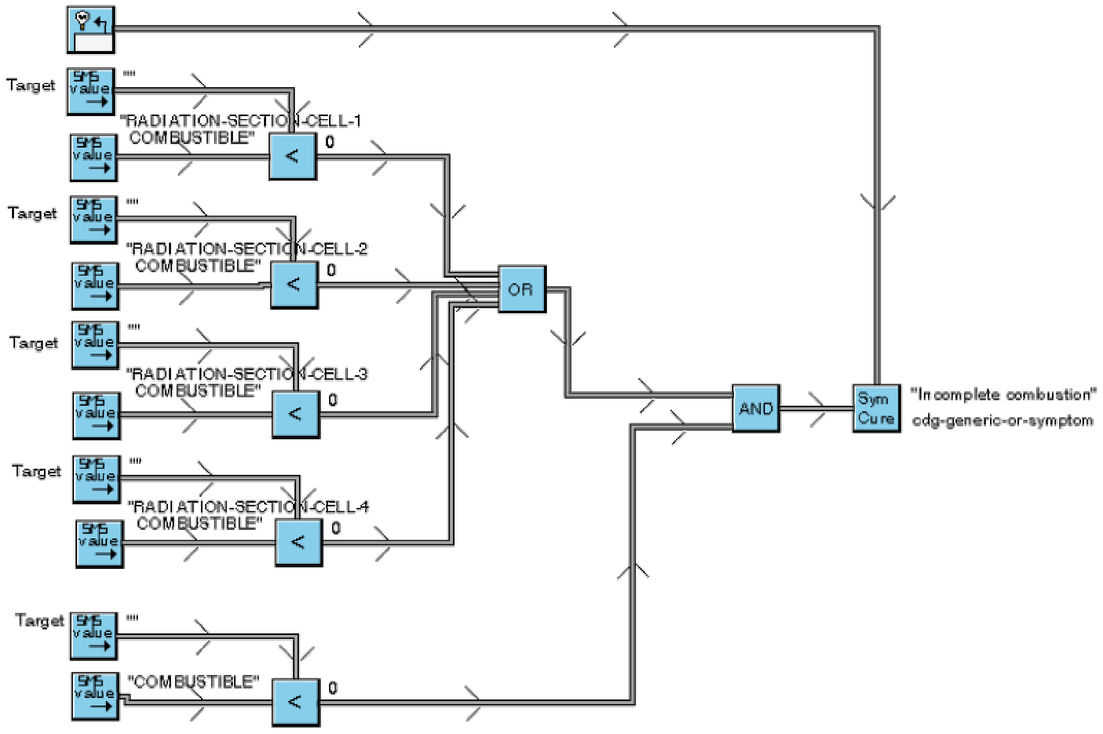
Asynchronous symptom and test result events are correlated using Specific Fault Propagation Models (SFPMs). SFPM is a fault propagation model that describes the propagation of fault, symptom and test events within and across specific domain objects. It is constructed at run time starting from the incoming events by appropriately combining the GFPMs and the Domain Map, just building enough event nodes to account for possible causes and effects of observed symptoms.

Relations among group of events are recognized based on the connectivity criteria in the SFPM such as the existence of a directed path or the fact that the events could be caused by common faults. Then the value of the incoming event is propagated in the SFPM to infer and predict the values of other events and to identify suspect faults. Suspect faults are identified by searching upstream from the incoming symptom and test result events with a true value in the SFPM.

Suspected faults are resolved by identifying appropriate candidate tests that, when executed, would provide additional information regarding those faults. Candidate tests are selected by searching downstream from the suspect faults in the SFPM. In the case of an automated test, a request is sent from SymCure to execute the automated test procedure. Otherwise, the test is displayed on the operator GUI for approval before execution. The candidate tests are ranked based on cost criteria such as resource use, disruptiveness, or the information value of a test. The test results are asynchronously input back for further correlation to reduce the number of suspect faults.

Based on the correlation of the test results, the suspected faults are either ruled out or concluded to have occurred. Diagnostic conclusions are outputs to the operator (see the paragraph “Operator Interface design and development”). Whenever a fault is concluded as a suspect or occurred, an appropriate mitigation actions specified for it can be executed. Similar to the test procedures, these mitigation actions can also be automated procedures or may require operator intervention. These mitigation actions can also be ranked based on criteria such as the failure-rate, the cost of fixing, or the cost of not fixing the faults.

The test and mitigation procedures were developed using Optegrity's OPAC (OPerations expert ACtions) graphical procedure development tool and a generic graphical block language, called ODP (Object Data Protocol), developed in this project.


Target
SMS value
""
"RADIATION-SECTION-CELL-1 COMBUSTIBLE"
"RADIATION-SECTION-CELL-2 COMBUSTIBLE"
"RADIATION-SECTION-CELL-3 COMBUSTIBLE"
"RADIATION-SECTION-CELL-4 COMBUSTIBLE"
"COMBUSTIBLE"
<
0
OR
AND
Sym Cure
"Incomplete combustion"
cdg-generic-or-symptom

Generic graphical code (ODP language) generating a symptom

The unified development and deployment environment provided by Optegrity allowed the incremental development of the application. Each single branch of the many fault models was independently tested using the simulation tools of Optegrity immediately after his creation, allowing to speed-up the application development, improving the software quality, and decreasing the time necessary to the final on-site tests.

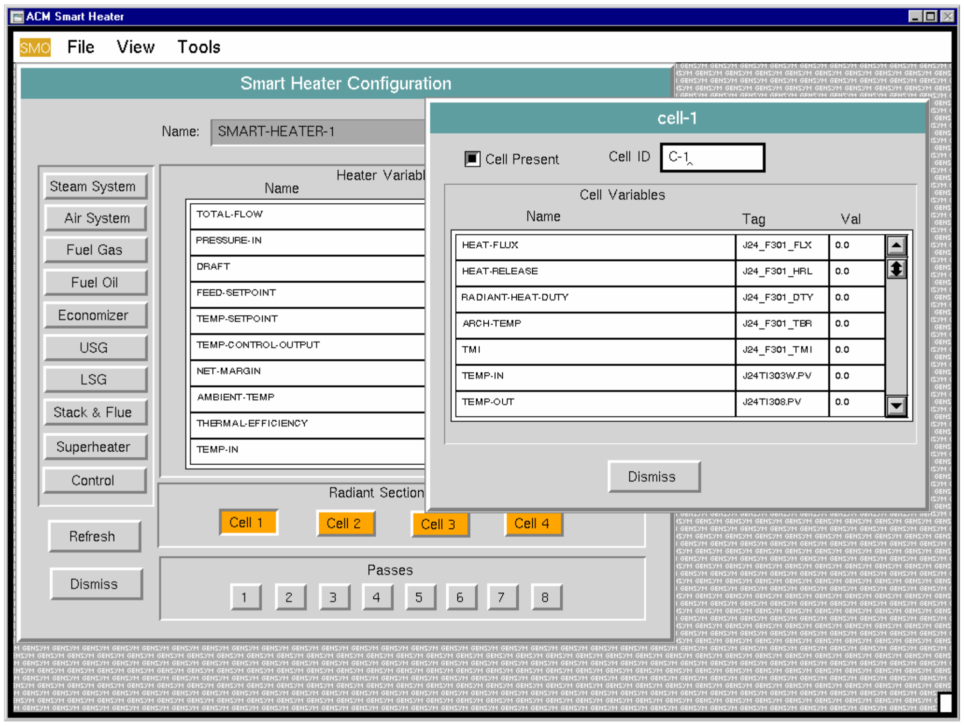
Operator Interface Design and Development

The output to be displayed to the operator consists of advisory messages, root-cause explanations, and action requests. Requirements for the Operator Interface were various, but the most important was the total integration with existing facilities. For such a reason it was decided developing a new interface, based on the Microsoft's component-container technology.

Using G2-ActiveXLink component (Gensym's bridge for integration with MS Windows operating systems), a custom interface has been developed in Microsoft environment. This interface is an ActiveX component, and it can be deployed in any ActiveX container. The value added by this software was very significant, because the end-user's GUI for the DCS is an ActiveX container, and the front-end interface of the Gensym's application could be perfectly embedded in the existing system. Moreover, there can be as many remote views as needed, and all of them are kept synchronized at all times, meaning a message acknowledged on one of these views will automatically update all views. All messages can be viewed, or filtered out via checkboxes (e.g. diagnostic messages, root-cause-messages, advisory-messages).

A valuable aspect of the generic smart heater is its reusability. For this reason particular attention has been paid to simplify the creation of a new object and its configuration. Configuration and maintenance is accomplished using Optegrity's traditional client/server graphical interface.

The heater smart object is an out of the box tool for most of its features, with preconfigured parameters and calculations, and built-in object classes covering any requirement. It includes also all process maps describing the configuration design, built-in smart sensors, as well as the event generation blocks for symptoms and tests. A total of 80 preconfigured faults identify the root cause of the problems and its relationship to the symptoms and the tests, while almost 240 messages provide requested information for diagnostic, root cause and operator guidance. To simplify the tag configuration task, a dedicated interface has been developed. Using this interface, tags are made independent from the communication interface with the field.



Name: SMART-HEATER-1

Heater Variabl

Name
TOTAL-FLOW
PRESSURE-IN
DRAFT
FEED-SETPOINT
TEMP-SETPOINT
TEMP-CONTROL-OUTPUT
NET-MARGIN
AMBIENT-TEMP
THERMAL-EFFICIENCY
TEMP-IN

[■] Cell Present   Cell ID: C-1

Cell Variables

Name	Tag	Val
HEAT-FLUX	J24_F301_FLX	0.0
HEAT-RELEASE	J24_F301_HRL	0.0
RADIANT-HEAT-DUTY	J24_F301_DTY	0.0
ARCH-TEMP	J24_F301_TBR	0.0
TMI	J24_F301_TMI	0.0
TEMP-IN	J24TI303W.PV	0.0
TEMP-OUT	J24TI308.PV	0.0

Tag configuration interface

Through the tag configuration interface it is also possible enabling or disabling each subsystem in the heater, according to the model of heater.

Using the configuration tools provided by the application, creating a new heater requires very short time. This simple process can be described in the following steps:

1. Adding a new instance of heater to an application is just matter of graphically dragging and dropping the prototype from a palette to the desired workspace,

2. Configuring the heater design by accessing the user templates to configure the number of passes, the number of cells and the subsumed. This step can be bypassed since the heater can automatically build itself based on the available tags.
3. Assigning the right tags to the new object by point and click from the list of tags and map these tags to the heater parameters. The most significant activity in adding a new heater to the application is identifying the tags in the control system, the configuration activity will require just few hours and it can be executed by non-specialized personnel, no specific training on the products is required.

Field Interface development

Data from the plant comes from a Honeywell TDC3000 DCS, but the application is interfaced with the field through the OSI PI plant historian database. For communicating with the PI database, a standard Gensym module (called *PI bridge*) has been used. The bridge interfaces PI database using its API (Application Program Interface) procedures.

The application, however, is completely generic, and can be connected also with any other system, without changing anything in the smart heater kernel. The bridge configuration is transparent to the users. The only piece of software to be changed is the bridge, the module in charge of communicating with the specific external device. All the rest is unchanged, also thanks to the configuration tools described in the previous chapter that encapsulate all the specific communication details.

The capability of the system to be interfaced to a widespread standard as OLE for Process Control (OPC), allows improving its portability, and deploying new smart heater applications in very short time.

Return on investments

To justify the heater smart object project a rigorous methodology was adopted, based on a cost benefit analysis study. This study comprised the development of the economic indices that can help the management to make an investment decision. This methodology can be summarized in the following:

1. Identifying the economic parameters for the cost/benefit analysis. These indices included the IRR (Internal Rate of Return), ENV (Equivalent Net Value) and payback period.
2. Identifying the cost of the components. This included the cost of Optegrity software, the cost of other modules, the application development and integration, and the cost of maintenance.
3. Identifying the tangible benefits for having the heater smart object. These benefits included credit resulting from saving fuel, credit from maximizing charge throughput, credit from increased steam revenues, credit from saving in maintenance cost, credit from increasing the on stream factor of the equipment.
4. Calculating the IRR based on 5 years depreciation with operating cost for maintenance of the application.

The results of these calculations were very encouraging. The IRR was in the range of 120% to 180% and the payback period was in the range of 3 to 8 months, depending on the type of heater, the application of the heater and the existing operating conditions.

Conclusions

In this paper we analyzed the problems related to the abnormal conditions and compared traditional, reactive approach versus proactive support provided by Abnormal Condition Management systems based on real-time expert systems technology.

We considered a case study describing an application recently implemented on a refinery for managing heaters. We described the advantages brought by such a system, the project evolution and the project implementation. Some information about the return on investment for the project was also provided.

As a conclusion we may affirm that investing in Abnormal Condition Management should be a prime consideration for many manufacturing operations. The gains for these investments can often exceed the gains for investments in advanced process optimization. An Abnormal Condition Management application that detects and helps to avoid abnormalities before they occur can add five percent or more to an operation's profits.

The qualitative benefits of Abnormal Condition Management applications go beyond incident avoidance; they can eliminate or minimize the impacts of abnormal process conditions that range from off-specification production to unexpected shutdowns to serious safety or environment incidents.

Bibliography

[1] "Abnormal Condition Management – The Missing Link between Sustained Performance and Costly Disruptions", Asish Ghosh, David Woll, ARC Advisory Group, March 2001.
[2] "Minimizing Process Disruptions and Sustaining Performance through Expert Systems Technology", Dave Siegel, National Petroleum Refiners Association 2001 Computer Conference , October 1-3, 2001, Dallas, Texas
[3] "A Generic Fault Propagation Modeling Approach To On-Line Diagnosis And Event Correlation", Gregory M. Stanley, Ramesh Vaidhyanathan, 3rd IFAC Workshop on On-Line Fault Detection and Supervision in the Chemical Process Industries, June 4-5, 1998, Solaize, France
[4] "Regole e procedure vive nei Sistemi Esperti: linee guida per formalizzare e condividere la conoscenza in applicazioni "intelligenti", Fulvio Roveta, Giornata di Studio sul recupero della conoscenza negli impianti industriali - L'automazione: tecniche di revamping e reverse engineering – Fondazione Aurelio Beltrami, 11 Aprile 2002 , Milano
[5] "Optegrity SymCure Modeling Guide", Version 3.4 Rev. 0, Gensym Corp., June 2001