COLLABORATIVE DECISION SUPPORT DURING PROCESS OPERATIONS USING HETEROGENEOUS SOFTWARE AGENTS

Y.S. Ng and R. Srinivasan*

Laboratory for Intelligent Applications in Chemical Engineering,
Department of Chemical and Environmental Engineering,
National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260
**Corresponding Author: Email: chergs@nus.edu.sg*

Abstract: This paper elucidates the methodology of employing agent-based technique for detecting and diagnosing faults in chemical processes. The proposed agent methodology has been shown to be able to effectively combine various fault diagnosis methodology and exploit the capability of parallel computing technology in the process plants. We also show that the integration and collaboration among heterogeneous methods is essential to overcome the numerous hurdles of effectively monitoring process transitions given its high level of complexness. Additionally, the framework shows promises in effectively managing information from various entities of a process plant, further enhancing the decision making of plant management. The framework is illustrated using a case study of the Tennessee Eastman Process.

Keywords: fault diagnosis, agents, parallel computation, decision support systems.

1. INTRODUCTION

Fault detection and diagnosis has become an area of primary importance in modern process automation. It provides the foundation for a fault tolerance and reliable system which minimizes the level of disruptions upon daily production. However, the advancements arising from the process systems engineering community have led to the introduction of numerous applications to support efficient process operations -data reconciliation, model-based control, optimization, state identification, monitoring, fault diagnosis, etc. As a result, human operators often encounter difficulties in managing and responding to the vast amount of information produced. Furthermore, with the increasing scale and complexity of process operations, especially non steady-state ones, there is a clear need to move beyond piece-meal solutions to a comprehensive, synchronized approach to support decision making. In Ng and Srinivasan, (2004b), we highlighted the need for integrating heterogeneous methods for collaborative plant-wide monitoring and supervisory control during process transitions. There, the integration of different fault diagnosis methodologies e.g: enhanced trend analysis, neural networks, dynamic principal component analysis (DPCA), self-organizing maps (SOM), and signals processing techniques- was proposed. The rationale of such an approach is based on the precept that a method that works well under one circumstance might not work well under another; the strengths of the various methods can be brought to bear on the problem and the drawbacks of the individual methods can be overcome through collaboration. In this paper, we outline the methodology to realize this integrated system in practice.

The organization of this paper is as follows: Section 2 provides a literature review of agent systems and some of their applications. Section 3 describes the framework of the proposed agent-based decision support system (DSS) for abnormal events management while Section 4 illustrates the application of the framework to a case study of Tennessee Eastman Process.

2. AGENT-BASED SYSTEMS

The term agent is defined as a computer system that is situated in some environment, and is capable of performing self-directing actions in this environment in order to meet its' design objectives (Wooldridge and Jennings, 1995). Distributed computing that is based on agent has been regarded as the next significant breakthrough in software development with capabilities to perform autonomous actions in open, dynamically changing environment. It offers opportunities to solve a complex problem collaboratively using heterogeneous methods across multiple platforms. Agent is generally characterized by its underlying knowledge/methods, and it interacts with other agents using a common agent communication language (ACL).

Agent-based computing has found roots in numerous fields since its inauguration in the early 90's. Some of its previous applications are presented here. Lefrancois and Montreuil (1994a,b) used agent concept to support decision making during the analysis and design of manufacturing system. Their agents are capable of cooperatively and interactively decide upon the actions to be taken, with a goal directed behavior and responsiveness to environmental changes. Qian et. al. (2000) proposed an agent based system for Computer Integrated Process Operation System (CIPOS). The agents, being implemented through object-oriented approach, integrate monitoring, process operation, and production management areas of a plant, resulting in an unified plant wide information update. Cho et. al. (2003) developed an agent-based management model to monitor network configuration. They accustomed their agents with different sets of rule-based reasoning systems to diagnose and recover known network faults. Applications of agent-based technique to the domain of supply chain management can be found in Julka et. al. (2002a,b). Julka et. al. (2002a) developed a framework to integrate various entities of the supply chain, e.g. enterprises, production unit, business portfolios and represent them in a unified fashion for enhanced decision support. They further illustrated their system through a refinery application (Julka et. al, 2002). Despite the broad disciplines of agent applications, none of the existing systems can adequately address the complex issues of managing fault diagnosis in the process industries.

3. COLLABORATIVE AGENTS FOR MANAGING ABNORMAL EVENTS

In this section, we describe a framework to effectively manage abnormal events in chemical processes – CAMEO (Collaborative Agents for Maintaining Efficient Operations). CAMEO was first proposed by Ng and Srinivasan (2004b). The authors reviewed numerous methods that are capable of performing control and supervision of process transitions, e.g. qualitative trend analysis, rule-based system, signal processing methods, neural-networks, statistical analysis methods etc, and emphasized the needs of moving beyond piece-meal solutions to a more comprehensive, integrated means as the complex facets of transitions preclude any single control and monitoring method from being effective. The rationale for an integration among heterogeneous methods is based on the precept that a method that works well under one circumstance (nature of process, nature of operation, scale, noise level, complexity) might not work well under another; the strengths of the various methods can be brought to bear on the problem and the drawbacks of the individual methods can be overcome through *collaboration.*

3.1 Agents environment

The CAMEO agent system is abstracted hierarchically into environment, host and agents. An agent environment in CAMEO is an artificial environment that make up of the process hardware, human personnel, and cluster of machines, where the resources of the machine clusters can be utilized interchangeably. The cluster can be as simple as the containment of computing resources from the local section of a process plant, or be extended to a group of pc clusters where all their processing resources can be relocated upon request, as in the case of heavy computing workload. Each agent environment might contain a number of hosts where each host is registered with its own agents. Each host also contains methods for searching of relevant agents, and for evaluating the hardware performance of a local machine, eg: number of processors, current workload, and suitability of allocating new agents. At any point of time, the host will hold a set of objectives to be fulfilled, eg: minimizing processing time, perform monitoring, diagnosis, perform fault recovery, reduce workload etc, which sets the priorities for the underlying agents upon time of execution. Each host will also keeps track of the flow of messages among different types of agents. In short, the host determines whether an agent is suitable to be relocated into it and keeps track of inter-host communication among agents to facilitate the process of debugging.

3.2 Agents as distributed objects

The agents in CAMEO are developed based on distributed object methodology. Distributed object methodology allows computing systems to be integrated across multiple platforms. The distributed object is not necessarily a complete application but rather a reusable, self-contained piece of software that can evaluate its own in-built logical parameters to decide upon its future actions to be taken. Each distributed object is aware of its domain of expertise to which they are specialized in, and contains a contact list of other classes of objects that they can interact with that is private to other objects. An agent in CAMEO can thus be described as having three main components- its object definition, methods containment, and contact list. The agents, being developed though distributed objects methodology, are also platform independence. They can operate autonomously on both client and server location with their operations unaffected through a unified communication language.

3.3 Agents entity

Agents are the primary entity in CAMEO with capability to perform autonomous actions in the agent environment. CAMEO incorporates six classes of agents, each of which addresses a different facet of transition operation. The different classes being defined are:

1. Data Management Agents
2. Operation and Control Agents
    2.1. Regulatory Control Agents
    2.2. Sequential Control Agents
    2.3. Alarm management agents
3. Supervision Agents

3.1. State identification Agents
    3.2. Monitoring Agents
    3.3. Diagnostic Agents
    3.4. Operator Performance Assessment Agents
4. Consolidator Agents
5. Fault Tolerance and Recovery Agents
6. Visualization and User Interface Agents

Data management agents provide the conduit to the process and its states for all other agents. Raw sensors data are read, de-noised, and reconciled by data management agents and the smoothed data is made available to other interested agents through broadcast. In this work, a windowed finite impulse response (FIR) filter (DSP committee, 1979) is implemented to filter out high frequency noise.

The *operation and control agents* aid the operator in executing and controlling the different steps of a process transition. Some examples of these follow: *Regulatory control agents* interact with the DCS and perform actions such as reconfiguring the controller settings based on the current state of the process. The *sequential control agents* coordinate among the discrete steps required for executing sequential operations. The *alarm management agents* help reconfigures the alarm management system to the current state and thus prevent alarm floods.

The goal of *process supervision* agents is to ensure safe operation. For effective process supervision, knowledge about the current process state is essential. Also, one way to prevent abnormal situations and make plant automation applications function appropriately is to make them cognizant of the domain of their applicability. These applications can then enable or disable themselves in specific modes and reconfigure themselves with the correct settings when different states have to be handled differently. In order to automate such switching, context sensitive information regarding the process state should be provided to the plant automation applications, which would then reconfigure themselves to the operating state. This is possible only if the different process states are previously known and their occurrence can be identified online. In CAMEO this role is performed by the *state identification agents.* The o*perator performance assessment agents* manage the log book of human operators and evaluate their performance: quality of product being manufactured, types of alarms generated, and response time during abnormal events. The agents can further increase the competitiveness of the employees by enabling their performance to be monitored constantly. Operator performance assessment agents will also facilitate plant managers in assessing plant employees by generating relevant reports automatically upon request.

Other supervisory agents are *monitoring agents*, which detect abnormalities, and *diagnostic agents*, which are responsible for diagnosing the root cause. Monitoring and diagnotic agents can incorporate the various methods such as qualitative trend analysis, neural networks, rule-based system, etc. Successful fault isolation is achieved by collaboration between supervisory agents and consolidator agents. Results from monitoring agents will be sent to consolidator agents to resolve possible conflicts generated from the multiple monitoring techniques that maybe applicable in any given state. Having solved the conflicts, the consolidator agents, will entrust diagnostic agents to locate the possible root cause(s) of the situations. Subsequently, further appropriate corrective actions will be planned by fault tolerance and recovery agents. This may require context specific tasks which will be performed by sequential control agents.

In this work, two monitoring methods have been implemented namely principal component analysis (PCA) and self-organizing map (SOM) through PCA-supervision agent and SOM-supervision agent. PCA is a statistical technique for modeling and exploring of process data. The PCA model constructed is combined with SPC technique for process monitoring (Jackson, 1991). Unsupervised neural networks, specifically self-organizing maps (SOM), have been gaining much attention lately for their ability to project high-dimensional data set to two dimensions while preserving the metric relations of the original data. In this way, the effort needed to visualize the process state during transitions can be reduced many fold as compared to traditional variables control charts approach when the process variables are colossal. SOM has been shown to be readily explainable, simple and easy to visualize. The evolution of a process during the transition and the corresponding states can be inferred directly from the U-matrix of SOM (Vesanto, 1999). The trajectory of the process can be used to deduce whether the operation is normal since a faulty operation will result in deviation from the normal trajectory. The SOM can also be used for automated fault diagnosis and root cause identification by linking it to cluster analysis and sequence alignment technique.

When multiple methods are used concurrently, conflict resolution is needed to arbitrate among the different solutions proposed by the different methods and provide one consolidated solution to the operator. In CAMEO, this role is performed by *Consolidator agents*. The consolidator agents contain the logic to enforce consistency among different agents and are the bedrock for the collaboration mechanisms. Other agents including supervision agents, operation and control agents, sub-tasking agents and visualization agents communicate their results to consolidator agents. These in turn use process state, historical performance, domain knowledge and other basis to resolve any conflicts that arise between agents. The consolidated conclusion regarding the normality of the process operation and the root causes of any deviation provide the basis for planning corrective actions.

The *fault tolerance and recovery agents* use process knowledge, available preplanned SOPs for specific situations, as well as historical records of corrective control actions to guide the process into a safe state to recover from an abnormal situation. The sequence of corrective measures generated by these agents can be implemented directly by the operation and control agents or communicated to the operations personnel through *user interface agents*.

A state specific context-sensitive graphical user interface is provided by the *visualization agents* in CAMEO. A more powerful visualization tool than the trend charts and other currently available techniques is needed so that plant personnel can visualize the progression of process more easily. The visualization agents incorporate methods for projecting high dimensional data onto a much lower dimensional grid and thus provide a means for plant personnel to visualize even long duration process transitions effectively. *User interface agents* gap the bridge between human-machine interactions, providing automatic email services to inform the relevant authorities regarding the events occurring in a plant, e.g: detection of fault, successful diagnosis of faults, state change operation, and request for responses. The SMTP mail server must be available to the user interface agents in order to send outgoing messages, or a pop-up message will be created on the local host otherwise. User interface agents also act as a medium between the users and operation and control agents to inflict user command towards the process operation units, eg: opening and closing of valves, starting up or shutting down of equipments, controllers parameters change etc.

Aside from the six classes of agents that are operational oriented, a new class of agent namely *sub-tasking agents* is required to facilitate their implementation. *Sub-tasking agents* are responsible for speeding up computation time by reducing the workload of the local host. Sub-tasking agents provide a means for inter-platform transport among agents. Agents that require high computing resources can be transported to other platform through s*ub-tasking agents*. Our *sub-tasking agents* conform to the MPI standard (Geist et. al., 1996) for portability and platform independence. We use MatlabMPI (Kepner and Ahalt, 2004) for inter-platform data exchange. MPI is a high performance parallel computing protocol for messages passing between serial processes. The completed results are then sent back to the sender (local-host) for further analysis or triggered other agents for further processing in the remote host.

The collaboration among the agents makes the framework capable of providing multi-faceted automation for monitoring the process performance and preventing human errors from occurring.

4. TENNESSEE EASTMAN CASE STUDY

The proposed framework is tested on the Tennessee Eastman process as proposed by Downs and Vogel (1993). The process consists of five major unit operations: a reactor, a product condenser, a vapor-liquid separator, a recycle compressor, and a product stripper. There are altogether 22 continuous process measurements, 12 manipulated variables and 19 composition measurements sampled less frequently. The control system used here is the decentralized PID control system of McAvoy and Ye (1994), as implemented by Singhal and Seborg (2001). The process faults, as proposed by Downs and Vogel (1993) are tested here under Mode 1 operations. All simulations are run for 60 hours with a 1 min sampling frequency. The faults are introduced at 720 min.

The reachable resources and their corresponding operating systems are first registered by the local host, the host for the TE process will be referred to as TEhost from here onwards for simplicity. The interactions among the different agents are shown in Figure 1. All of the agents are first initialized and registered by the TEhost. The TE process simulator simulates the dynamics of the process while the decentralized control system of McAvoy and Ye (1994) will make up the *operation and control agent* in this case study. The *data management agent* conduits the process of data acquisition from hardware (simulator) and implements a FIR lowpass filter to remove noise before the data are further reconcile to remove redundancy before being distributed to *operations and control agent* and *consolidator agent*. Having the filtered data collected, *operation and control agent* implements feedback control actions to maintain the controlled variables at set point.

The *consolidator agent* first initializes two *supervision agents* locally, namely PCA-supervision agent, and SOM-supervision agent. The consolidator agent in TEhost is given the objective functions of not having more than one monitoring agent, and keeping the processing cycle time less than sensors sampling frequency. As the SOM-supervision agent is recognized as more computational intensive by default, the SOM-supervision agent is forwarded to sub-tasking agents. The subtasking *agent* then issues an auction to all remote-hosts, broadcasting the availability of a task to be completed by TEhost. Each remote-host then evaluates its current host status and replied to the auctioneer with the availability of processors and their latest status. The SOM-supervision agent is then allocated to the remote host that placed the highest bid-value, where bid-value=

$$\sum_{1}^{n}(connectivity_n x CPUperfomance_n x CPUclockspeed_n)\,.$$

The evaluation of bid is based on the consideration of three factors, 1: the network connectivity, which limits the rate of data transfer, 2: the processors' performance, which shows the free resources available, and 3: the processors' clock speed, which determines the efficiency upon computation of a task. The entire state of the agents, and its corresponding data structure, are then transferred to the remote host through MatlabMPI for execution, thus reducing the workload of the TEhost.

The *state-identification agent*, having identified the current mode of operation, will coordinate with the monitoring agents to identify the correct monitoring models for the monitoring agents. The scores plot of the PCA-monitoring results for IDV(2) (B composition change, A/C ratio constant) is shown in Figure 2. The fault is detected at 756 min when the data violates the 99.9% confidence limit. The fault is further confirmed by SOM-monitoring agent at 758min, as shown in Figure 3. The early detection of the fault kicks-off the execution of the diagnosis agents. The current framework implements the SOM-based diagnosis methodology as proposed by Ng and Srinivasan, (2004a). The method analyzes the signatures of a fault through syntactic pattern recognition approach and identifies possible root causes from a fault database. The variables residuals generated through SOM-based methodology is shown in Figure 4. The consolidator agent, having the diagnosis results identified, alerts fault tolerance and recovery agent to implement preplan SOPs for IDV(2).The *consolidator agents*, acting as a mediator, further interacts with *visualization and user interface agents.*

Visualization agent provides the graphical aide to the human personnel by presenting the lower-dimensional images to amplify process visualization, while *user interface agent* generate emails automatically to inform relevant authorities of the abnormal event. The details of the abnormal events are also stored in the operator log book by the *operator performance assessment agent*. The operator performance assessment agent keeps track of the response time of a particular operator to warning messages, their decision in responding to abnormal events, and utility charges/product information under their supervision.

5. CONCLUDING REMARKS

This paper presents a framework to efficiently manage a process plant through an agent-based approach. The proposed approach provides a means to integrate existing software and exploit the features of parallel computing in process plant. The proposed agent-based architecture can monitor a plant more effectively than any piece-meal method and pave way for a collaborative approach to solve a complex problem of fault diagnosis in transient processes. The testing on Tennessee Eastman process revealed an improved efficiency in data and information management in comparison to any conventional practices. The proposed method has also been found to be more effective in fault diagnosis, resulting in higher rate of successful detection and lower rate of false alarms, as the strengths of the various methods have been combined with their drawbacks overcome through collaboration. Current work is oriented towards the integration of additional monitoring agents and implementation of results fusion to enforce consistency.

REFERENCES

Cho, K.J., S.J. Ahn, and J.W. Chung, (2003). A study on the classified model and the agent collaboration model for network configuration fault management, *Knowledge-Based Systems* ,**Vol 16**, 177-190.

Downs, J.J., and E.F. Vogel, (1993). A plant-wide industrial process control problem, *Computers and Chemical Engineering*, **Vol 17(3)**, 245-255.

DSP Committee (eds), (1979). Programs for digital signal processing, *IEEE Press*, New York.

Geist, A., W. Gropp, S. Huss-Lederman, A. Lumsdaine, E. Lusk, W. Saphir, T. Skjellum, and M. Snir, (1996). MPI-2: Extending the Message-Passing Interface, In *Euro-Par '96 Parallel Processing*, (L. Bouge, P. Fraigniaud, A. Mignotte, and Y. Robert. (Ed)), number 1123 in Lecture Notes in Computer Science, 128-135, Springler Verlag.

Jackson, J.E., (1991*). A user's guide to principal components*, John Wiley & Sons, New York.

Julka, N, I. Karimi, and R. Srinivasan, (2002). Agent-based supply chain management - 2: a refinery application, *Computers and Chemical Engineering*, **Vol 26**, 1771-1781.

Julka, N., R. Srinivasan, and I. Karimi, (2002). Agent-based supply chain management - 1: framework, *Computers and Chemical Engineering*, **Vol 26**, 1755-1769.

Kepner, J. and S. Ahalt, (2004). MatlabMPI, *Journal of Parallel and Distributed Computing*, **Article In Press**.

Lefrancois, P., and B. Montreuil, (1994a). An object-oriented knowledge representation for intelligent control of manufacturing workstation, *IEEE Transactions*, **Vol 26(1)**, 11.

Lefrancois, P., and B. Montreuil, (1994b). An organism-oriented modeling approach to support the analysis and design of intelligent manufacturing system, *Journal of Intelligent Manufacturing*, **Vol 5**, 121.

McAvoy, T.J., and N. Ye, (1994). Base control for Tennessee Eastman problem, *Computers and Chemical Engineering*, **Vol 18(5),** 383-413.

MPI forum, available at http://www.mpi-forum.org/

Ng, Y. S. and R. Srinivasan, (2004a). Monitoring of Distillation Column Operation through Self-organizing Map, Presented in *7th International Symposium on Dynamics and Control of Process System (DYCOPS),* Massachusetts, USA, July 5 – 7, 2004.

Ng, Y.S. and R. Srinivasan, (2004b). Transitions in the Process Industries: Opportunities and Prospective Solution, Presented in *IEEE International Symposium on Intelligent Control (ISIC)*, Taipei, Taiwan, Sep 2 – 4, 2004.

Qian, Y., Q. Huang, W. Lin, and X. Li, (2000). An object/agent based environment for the Computer Integrated Process Operation System, *Computers and Chemical Engineering*, **Vol 24**, 457-462.

Singhal, A., and D.E. Seborg, (2001). Matching patterns from historical data using PCA and distance similarity factors, *Proceedings of the American Control Conference*, 1759-1764.

Telmoudi, A. and S. Chakhar, (2004). Data fusion application from evidential databases as a support for decision making, *Information and Software Technology*, **Vol 46**, 547-555.

Vesanto, J., (1999). SOM-based data visualization methods*, Intelligent Data Analysis*, **Vol 3**, 111-126.

Wooldridge, M., and N.R. Jennings, (1995). Intelligent agents: theory and practice, *The Knowledge Engineering Review*, **Vol 10(2),** 115-152.

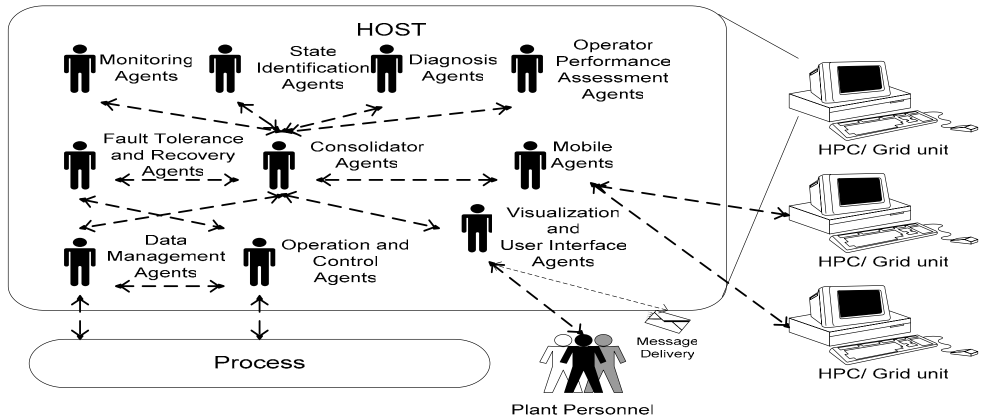

Figure 1: Framework of CAMEO

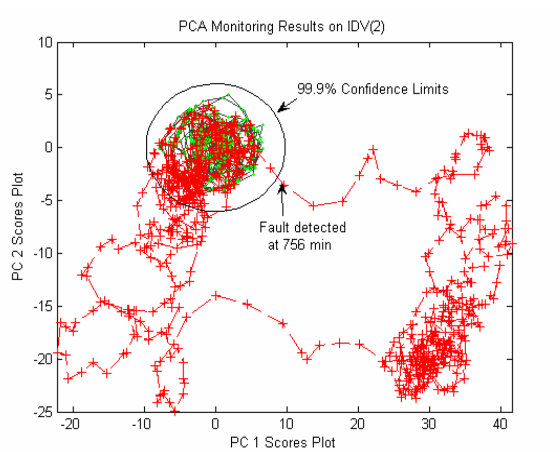

Figure 2: PCA monitoring-agent result: scores plot of principal component analysis

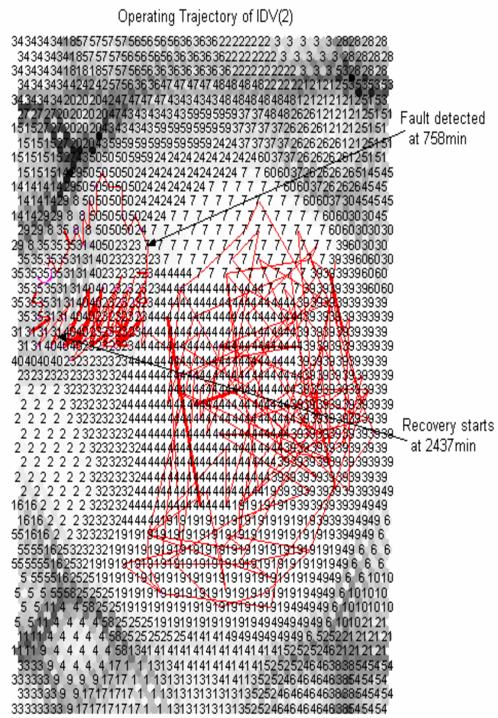

Figure 3: SOM monitoring-agent result: process operating trajectory on U-matrix of self-organizing map

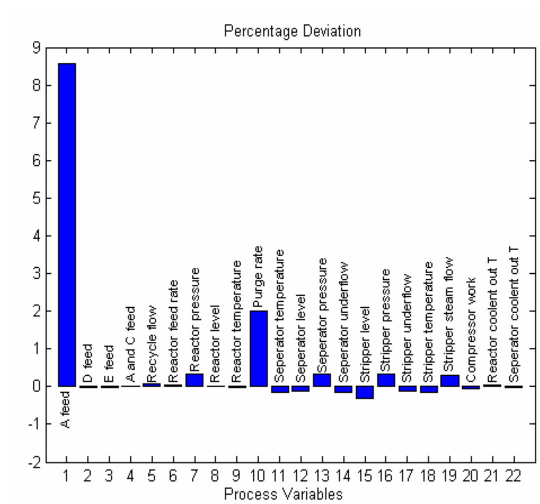

Figure 4: Variable residuals based on SOM