# FAULT DIAGNOSIS IN BATCH PROCESSES

N. J. SCENNA

*UTN.FRR -Zeballos 1341- 2000 Rosario- Argentina*
*INGAR(CONICET) -Avellaneda 3657- 3000 Santa Fe- Argentina*

**Keywords**: Fault Diagnosis, Batch Processes, Hierarchical Systems, Modular Approach.

**Abstract**

**The aim of this paper is to perform an analysis of the fault diagnosis problem in a single mono-product batch chemical process. As a derivation of the analysis of the principal aspects of the problem, some guidelines to implement an efficient algorithm (Modular Hierarchical Fault Diagnosis System -MHFDS-) will be exposed. An application to a single batch process is presented.**

## 1. Introduction

Batch operations are characterised by time evolving variables and parameters. The normal state can be associated to a set of desired trajectories that can be labelled as normal evolutions. Thus, in addition to non-linear models, we must deal with time evolutions and the necessity of supervising appropriate initial conditions.

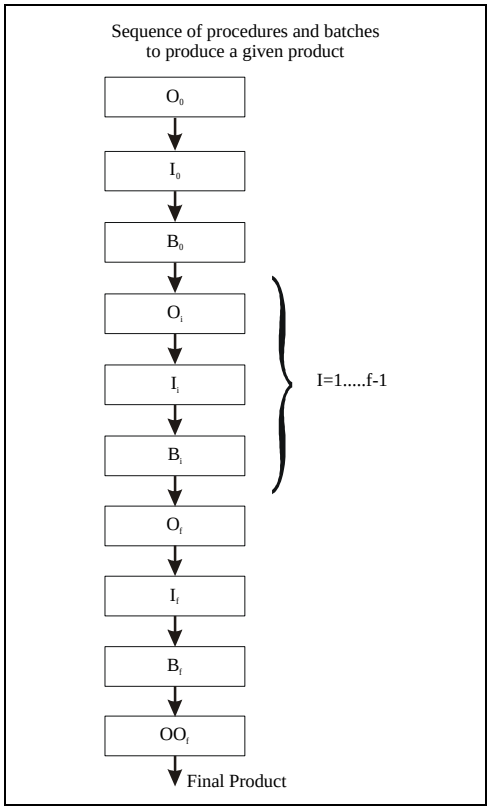



Fig. 1. Temporal Blocks Decomposition

A diagnostic problem is characterised by a set of observations to be explained. When these observations manifest discrepancy with the expected behaviour, we define the system as having a faulty state. Thus, given a set of symptoms (temporal observations) we must deduce (diagnose) which fault/(s) can explain the abnormal behaviour (observed symptoms). Under this context, fault diagnosis in batch processes can be viewed as an interpretation problem of sensor patterns using useful symbolic abstractions, in a multi-variable, non-stationary environment.

To simplify our problem, we will not analyse multiproduct batch processes. This hypothesis facilitates the decoupling of the temporal blocks. In Fig. 1 we represent a sequence of actions beginning with a block of operative actions $O_0$ , which includes the set of tasks to be performed (according to the normal process recipe) before the equipment $E_0$ starts the batch $B_0$. Due to these actions, plus the status of the material streams (row materials for example) and energy "streams" (temperature of certain utilities for example), the set of initial conditions ($I_0$) of the batch $B_0$ is defined. Then, the block $I_0$ containing a set of normal or abnormal initial conditions is incorporated. After finishing the first stage $B_0$, a new set of actions must be carried out (manually or by a PLC) according to the process recipe. Thus, stage by stage, (represented by a generic sub-index i) new blocks are added to represent the entire evolution along the time horizon of the process.

Finally, we arrive at the last stage (f), after which the batch process is finished. Then, we define a block $OO_f$, which represents, according to the process recipe, the set of final actions to be performed to obtain the product in normal state (see Fig. 1). Following the above process division, two principal modules can be differentiated. While the set of modules $O_i$ and $I_i$ are strongly dependent on the process context, modules $B_i$ represent conventional batch equipment, which in general (but not always) can be modelled using deep (fundamental) knowledge (for example by a set of ODEs plus a set of constrains defined by the problem case).

## 2. Problem Solving Methodology. A Hierarchical Hybrid Modular Approach

System theory tells us that complex problems or tasks (like fault diagnosis) can be analysed and decomposed into a smaller set of sub-problems. This is a common strategy for the construction of Fault

Diagnostic Systems (FDSs). In fact, as it was explained since batch processes can be described as a set of procedures following a given recipe, we can arrange a set of temporal linked blocks representing a set of tasks. Then, modularity can be easily implemented following the task sequence associated to each one of the temporal modules. In fact, each batch unit temporally works independently of the others during the sequence of tasks. Thus, given the initial condition set $I_0$, after finishing the first task, the next operative actions and the next initial conditions must be satisfied to start the following batches, up to the arrival to the final product, as it is indicated in Fig 1. It is very important for the model to reach the capability of identifying which goals can be achieved by user actions (or not achieved due to wrong actions). Thus, a list of the operator (or computer) actions that must be performed to achieve each desired goals must be provided. From the teleological point of view, the model must be focused on the goals assigned to the system by the designer. The capability of the model to contemplate human errors (that have been recognised as the major source of faults specially in batch processes) is then of paramount importance. We have above described that two different types of models must be affronted. For modelling all tasks (or faults) associated to modules $O_i$ and $I_i$, we need to emphasise the teleological and functional approaches, covering the process knowledge (topology for example) and the set of discrete actions to be performed among the batch process stages (recipe). On the other hand, each of the continuous (or semi-continuos) unit operations (pumping, batch reaction, filtration, etc.) activated during the batch process can be modelled as a particular temporal evolution. Thus, we can implement for each unit, ($B_i$ module) a specific Fault Diagnostic Subsystem. For these specific units, we must emphasise deep knowledge rather than empirical knowledge.

*PCBs Definition. Qualitative Analysis*

Fault Diagnosis methods using qualitative approaches and particularly SDGs are well-known and widely used. In a SDG, the value of each node represents the qualitative state of a given process variable: normal (0), high (+1) or low (-1). Each arc represents the influence of a variable (initial node) on another variable (final node). A direct relation (both nodes deviate in the same direction) is shown by (+1) gain, the inverse by (-1) and (0) shows no relation. Therefore, the gain $G_{i,j}$ corresponding to the arc from the node i to node j is a qualitative indicator of the direct effect on j due to a perturbation on i, and can be defined according to the following expression (Iri *et al.*,1979):

$$G_{ij} = \mathrm{Sign}\left.\left(\frac{\partial X_j}{\partial X_i}\right)\right|_{Xn}$$

where $\boldsymbol{Xn}$ is a vector representing the normal evolution. For batch processes, the use of only one digraph during the full time horizon is not good enough. In fact, gains are functions of both process state ($\boldsymbol{Xn}$) and topology, now time functions. To solve this problem, we can divide the entire process evolution in a set of special intervals or Pseudo Continuous Blocks (PCBs). A PCB is a period along which the batch process can be modelled with only one SDG. In other words, along a PCB the gains are qualitatively constant. There will be as many PCBs (and SDGs) as it is necessary to cover the entire time horizon. Gains can change due to topology variations, operative procedures, or due to the process evolution. For example, if a bioreactor operation changes from batch mode to perfusion mode, a new input flow appears. This change must be reflected in the corresponding SDG; then, another SDG is needed.
Gains can be calculated considering the process model (Iri *et al.*, 1979). Chemical processes, in general, can be represented by a system of first order differential equations and a system of algebraic equations:

$$\frac{dX}{dt} = f(X) \quad , \quad h(X) = 0$$

Hence, the linearisation of $\boldsymbol{X}$ around $t_0$ (just before fault activation) permits to express the gain as:

$$G_{ij} = \mathrm{Sign}\left.\left(\frac{\partial f_j}{\partial X_i}\right)\right|_{Xn}$$

From the SDG, we can obtain a rule set describing the fault propagation characteristics. For example, consider the system shown in Fig. 2. The mathematical model is:

$$A\frac{dL}{dt} = F_1 - F_2$$

$$F_2 = K\sqrt{L}$$

The remaining gains can be obtained from algebraic equations by implicit differentiation.

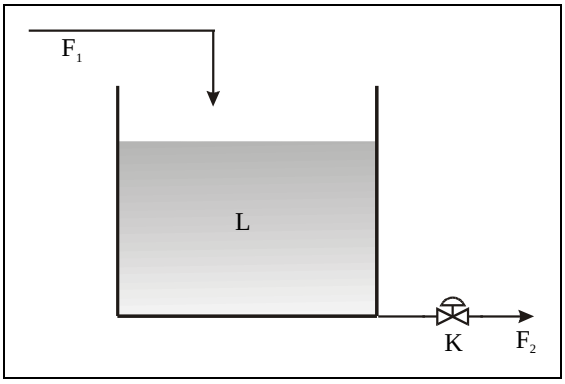
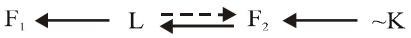



Fig. 2. A Single Tank

The following is the corresponding SDG.

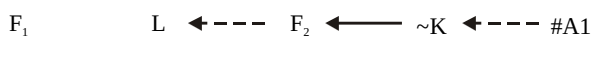



Solid arcs represent positive gains, whereas the dashed arcs have negative gains. The mark ~ indicates that *K* is a non-measured variable. Under the assumption that a single fault is the source of all disturbances, this fault affects initially a single node in the SDG, this node is called *root node*.

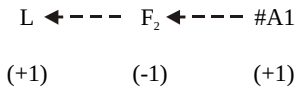
Then, the fundamental premise is that *cause and effect linkages must connect the root node to all the observed symptoms*.

For instance consider the fault #A1: a partial valve blockage. The SDG now includes a new node representing the studied fault. Due to this fault, a decay on *K* is produced, so the gain for the arc from #A1 to ~*K* is -1.

This stage is called *fault modelling*, it determines how the fault will propagate along the SDG. So, we have:




Tarifa and Scenna (1998a-b) proposed a method to obtain rules from the SDG by qualitative simulation, based on the previous one proposed by (Kramer and Palowitch, 1987), but with some modifications. For this example, qualitative simulation output is only one interpretation:




Note that the node $F_1$ is not included in the interpretation because it belongs to the sub SDG that it is not affected by the studied fault. Then, the corresponding rule is (Tarifa and Scenna, 1998a-b):

IF

{($F_2$ = -1) AND ($L$ = +1) AND ($F_2$ is abnormal before $L$ is)}

AND

{($F_1$ = 0) AND (NOT($F_1$ is abnormal before $L$ is))}

THEN

#A1 may be the fault.

The first two propositions belong to the fault pattern. They verify that $F_2$ is low and $L$ is high. The third proposition considers the symptoms order or symptoms sequence. The following proposition checks that $F_1$ is not being affected by the fault, then $F_1$ must be normal. Finally, the last proposition verifies that the arc from $F_1$ to $L$ is inactive. This proposition may be redundant, but it is included to maintain the symmetry, which is important using fuzzy logic. Formally, the Rule Set (RS) rendered by the compilation step can be expressed as follows:

$$RS = \{ R_1, R_2, \ldots, R_{NR} \}$$

$$R_i / \{ A_i \rightarrow C_i \}$$

So, RS is a set of rules $R_{ib}$ (one for each potential fault cluster of the system). Each rule has an antecedent $A_i$ and a consequent $C_i$. This consequent represents the cluster (set of faults with the same pattern) assigned to this rule. In this way, after the off-line stage, a real time expert system can be implemented.

In an attempt to consider the fault propagation dynamics, Umeda *et al*., (1980), extended the SDG interpretation (modelling) and the Gain definition for a dynamic system. In fact, they represented the ODEs system describing the dynamic behaviour (and the set of algebraic equations) of the process in a discretised form:

$$X_j^{m+1} = X_j^m + f_j(X_1^m, X_2^m, \ldots, X_n^m)\,\Delta t + \varepsilon_j$$

$$X_j^m = X_j(t_m) \qquad t_m = m\,\Delta t \qquad m = 0,1,2,\ldots,M$$

$$h_j(X_1^m, X_2^m, \ldots, X_n^m) = 0$$

where M is the number of intervals, and $\in_j$ is the truncation error. This discrete model can be represented with a spatial SDG, where a node is assigned to each discrete state of each variable. All nodes pertaining to time ($t_m$) form the active SDG for that time. Applying the gain definition, the arc gain $X_i^{m-1}$ to $X_j^m$ is now (Umeda, 1980):

$$G(X_i^{m+1}, X_j^m) = \mathrm{Sign}\left( \frac{\partial X_j^{m+1}}{\partial X_i^{m+1}} + \frac{\partial f_j^{m+1}}{\partial X_i^{m+1}}\,\Delta t \right)\Big|_{Xn}$$

In Fig. 3, the spatial SDG for a single equipment (a tank) with a variable operation mode is shown, because we suppose that at a given time (t*) a second discharge line (V2) is opened.

The spatial SDG for this process, according to the above explained criterion (left) contains internal arcs (for each given time interval) and temporal bridges between SDGs (between successive time intervals). Umeda *et al.* (1980) utilized their model to explain the fault propagation dynamics in continuous processes; but they concluded that, for high sample frequencies, this method becomes inefficient due to the high number of SDGs that must be analysed. On the other hand, if the sample frequency is low, the model may not track the real evolution. Moreover, they used an on-line search along the interconnected SDGs -instead

of a set of compiled rules-, which demanded too much computation (Iri *et al*., 1979).

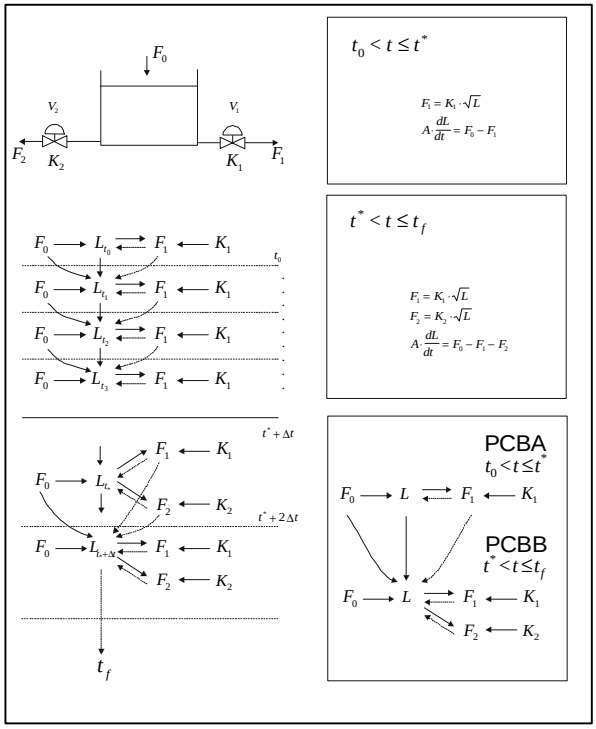



Fig. 3. Dynamic Signed Directed Graph (SDG).

*Some Aspects About Time Granulation and Model Properties.*

As it was remarked above, since we propose to represent the full time horizon by a series of PCBs, by definition the "global" SDG of the process can be obtained by the "union" of the successive PCBs through the differential variables (time bridges). Differential variables are those appearing on the left side of the differential equations, the other variables are algebraic ones. The graph model so obtained is very similar to the one used by Umeda *et al.* (1980). However, here, instead of changing at each time step the valid SDG, and to perform an on line search, we consider that successive SDGs are different only when some topology change or gain change is produced. In the above figure, the two PCBs for the process are indicated (after "compiling" equals SDGs) -different SDGs are produced when a topology change or a gain sign change appears-. With this procedure we extend the implicit linearization (SDG construction) to the temporal domain, in a sort of qualitative gain scheduling. This technique is well known in control theory. It is often difficult to find continuous control laws that are useful in all the relevant regions of a plant´s parameter space. Knowing how the dynamics changes according to the operating conditions, it is possible to implement a piecewise controller using different control laws according to the plant state. This principle is known as "gain scheduling" (Shama and Athans, 1990; Jacobs and Jordan, 1993). *Essentially, the entire plant evolution (dynamics) is divided into several operating zones covering the full operating range*. Each zone is characterized by a time-invariant (linear) approximation of the plant. Between adjacent or time successive operating zones, gains are interpolated or scheduled. Thus, "local or temporal" models are used in a sort of multi-modelling cooperative architecture.

On the other hand, this analysis (PCB division) is performed off-line. For each PCB a pre-processing step (qualitative simulation) gives us the set of rules describing the normal or faulty states to be used in online mode by the ES, as it was previously explained. Thus, each SDG has a time interval associated in which is valid; once this interval is expired, the following SDG is activated, and so on until the full time horizon is covered (see Tarifa and Scenna, 1999). Of course, one of the problems using SDGs is the necessity of a process model. In general, the diagnostic system is restricted by the model limitations. To explore other alternatives we can use the same strategy employing empirical models.

## 3. System Architecture

Here we postulate that the above procedure (*the division of the entire plant evolution in several operating zones covering the full operating range)* is a good choice for implementing a Modular, Hybrid, Fault Diagnostic System (MHFDS) for batch processes. So, local (temporal) models can be used for process modelling in a sort of multi-modelling cooperative architecture. We need a control system switching from different valid models as functions of time, the state of the process, etc. *(this naturally implies a hierarchical strategy).* The major attraction of this technique is the decomposition of the problem (task) in a set of sub-problems (sub-tasks) each one being easier to solve. Indeed, each PCB can be solved with an appropriate model (SDGs, Neural Networks (NNs), etc.). Figure 4 shows the FDS architecture. Since for many batch processes is so difficult to write an appropriate model, a general approach can be based on NNs. Nevertheless, analytical models must be used (for example SDGs) whenever possible.

In Fig. 5, the proposed system architecture for our MHFDS for supervising the entire process is indicated. We have an Expert System with different knowledge bases in which is encoded all the information to carry out the control actions, linking in real time all the sub-models reasoning lines. In fact, we have a sort of hierarchical multiple-model cooperative architecture in which the first hierarchy (the ES is the control level) encapsulates all the knowledge to link sub-models among them. Each one of the sub-modules encodes the necessary knowledge to perform the diagnostic task in each stage of the batch process. So, the control hierarchy contains rules with the information about the potential batch states, the fault modes propagation among non adjacent stages, and all the relevant information for the faults identification process.

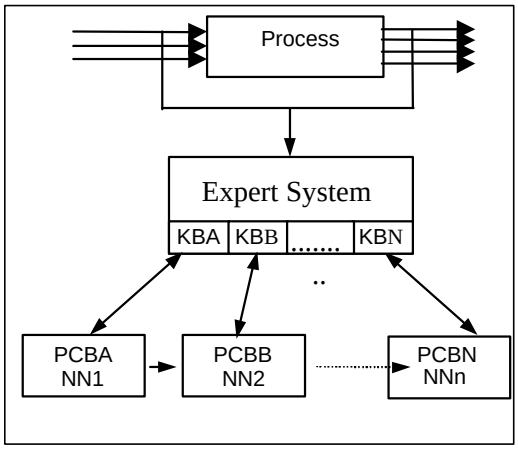



Fig. 4. A Modular Hierarchical Approach

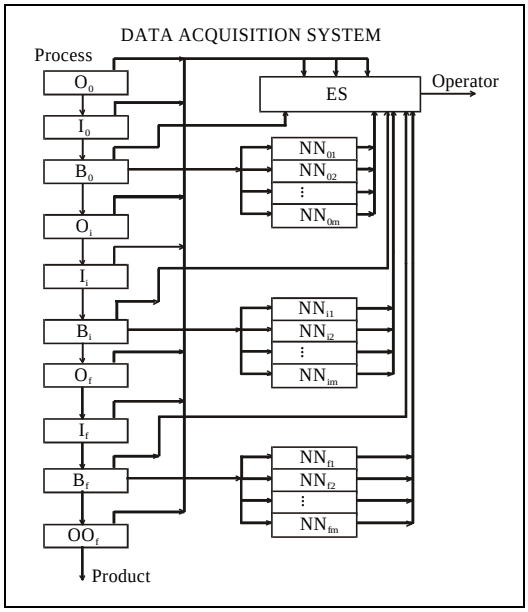



Fig. 5. The System Architecture

It is clear that the identification of each block, and inside each module ($B_i$), the identification of each sub-module (PCB) is a task that must be accomplished by the control level (ES). For this target, not only the absolute time (time intervals) is taken into account. Also, there exist state variables that must lie inside certain normality bands, which can be associated to the transitions between modules or sub-modules (PCBs). Finally, after a block is executed, the control system can reset the clock to begin the new time scale from the beginning of each module. Summarising, summed to the specific process knowledge and the one associated to the sub-models outputs, the ES must contain the necessary knowledge to identify the characteristics of the gain changes during the process evolution, the corresponding PCBs changes and the time in which the changes occur.

So, in addition to modeling blocks $I_i$ and $O_i$, the ES tasks in the hierarchical architecture shown in Fig. 5 are:

a) To handle the transition times and the interactions among submodels.
b) To define the NN (or sub-model) which must be activated, after the detection of an abnormal situation.
c) To interpret the outputs of the NNs in order to identify the origin of the detected fault.
d) To advice the operator on every abnormal situation and to inform the origin of the detected fault. The ES presents a ranking of possible faults candidates according to their associated probabilities. This information must be presented to the operator as friendly as possible, in order to avoid erroneous interpretations.

## 4. An example. A Batch Process

Figure 6 shows a simplified diagram of a typical batch process composed of a reactor, a distillation column and some storage tanks. Here, we can identify the different blocks according to the process recipe, following the general schema indicated in Fig. 1.

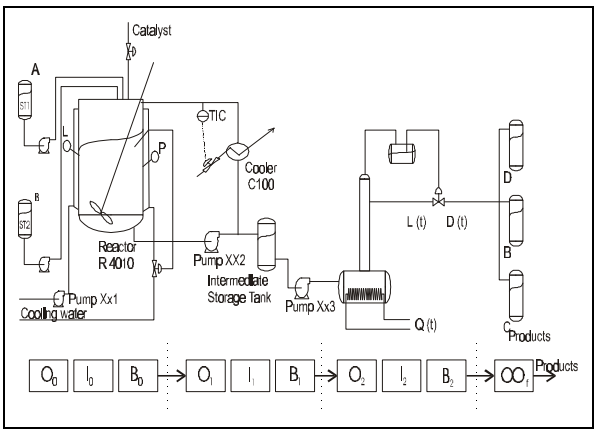



Fig. 6. The Batch Process Flowsheet

To implement a MHFDS, we must define the temporal modules and the models for each one. The actions to be performed by the PLC (or the operator) to adjust the initial conditions for the reactor, will define the module $O_0$. For example, the sequence of procedures to charge the reactants to the reactor, the set of valves and indicators that must be fixed in an initial position, etc. In addition, after all this initial actions and states of the process are encoded in the first module above mentioned, we can define the block $I_0$ , which stores all the information to decide if the given initial process state pertains or not to the set of normal initial conditions to obtain a good batch.

*The model for supervising operating procedures*

Operating procedures, as pointed out above, contains the information about the process and about the plant in which the batch process will be carried out. The set of intermediate modules $I_i$ or $O_i$, can be easily represented by an event Tree, which in turn can be easily encoded in a IF-THEN format rule. Given this knowledge representation (normal procedures), all the other knowledge to be captured is the knowledge about faulty actions that can be performed in each of the sequential programmed tasks (see Scenna, 2000).

In real time, and based on the measurements and the status of the process (valve opening status, signals to control loops, etc.) the problem is to infer if the process is conducted to an abnormal situation. If this condition is confirmed by the MHFDS (for example due to a wrong action of the operator) it activates an alarm and/or displays a window to inform the abnormal action and the fault origin.

The knowledge extraction (expert knowledge) can be performed using methods of Knowledge Engineering. Knowledge elicitation typically requires several cycles in which the expert operator is systematically guided through "abnormal" process scenarios (and event or operator actions) and motivated to provide an explanation. In this way, tools of Reliability Engineering like CCA (Cause and Consequence Analysis), HAZOP (HAZard and OPerability analysis) and FMEA (Failure Mode and Effect Analysis) can be applied (Henley and Kumamoto, 1981; Himmelblau, 1978).

In Fig. 7, the simplified event chain for the block $O_o$ corresponding to the start-up of the reactor is shown. The knowledge obtained after performing a HAZOP analysis can be represented by rules with the "IF..Then.." format. So, we obtain a set of rules like the following:

**IF** ((Level_R4010 is high) AND (Pressure_PI1030 is normal) AND (Temperature_TR1010 is normal) AND (RPM is normal) AND (time-t is included in the fuzzy interval $[T_0, T_1]$)

**THEN** (The Fault is A1).

The rule set must cover all the normal sequence of actions and the abnormal situations. The problem here is the data acquisition system and the process instrumentation. In fact, not all the variables are measured in on-line mode, (certain valve positions for example could be not monitored). Thus, we can only process a simplified event tree. All the other potential faults (which cannot be detected by the system at a given stage) must be modelled as faults to be identified during the evolution (time horizon) of the corresponding next stages.

As above explained, time spent in some tasks is very important from the process safety point of view. For example, certain actions have a concrete time in which they must be carried out. Thus, an absolute time (reference landmarks) must be used to check these actions. In this way, each task is associated to a normal time interval in which it must be carried out. However, to avoid the same difficulties originated due to the use of the rigid evaluation employing binary logic in conventional fault diagnosis systems, it is convenient to introduce fuzzy logic for time manipulation. In other words, when an action must be performed at a given time T, we permit an interval of times whose membership degree to the normal and to the abnormal sets are calculated according to the rules of fuzzy logic. Indeed, we can associate to each transition time, a set of state variables, whose values can be related to a normality band, associated to the state transitions. In this way, we can supervise the transition time among PCBs independently of time (using a set of process variable values).

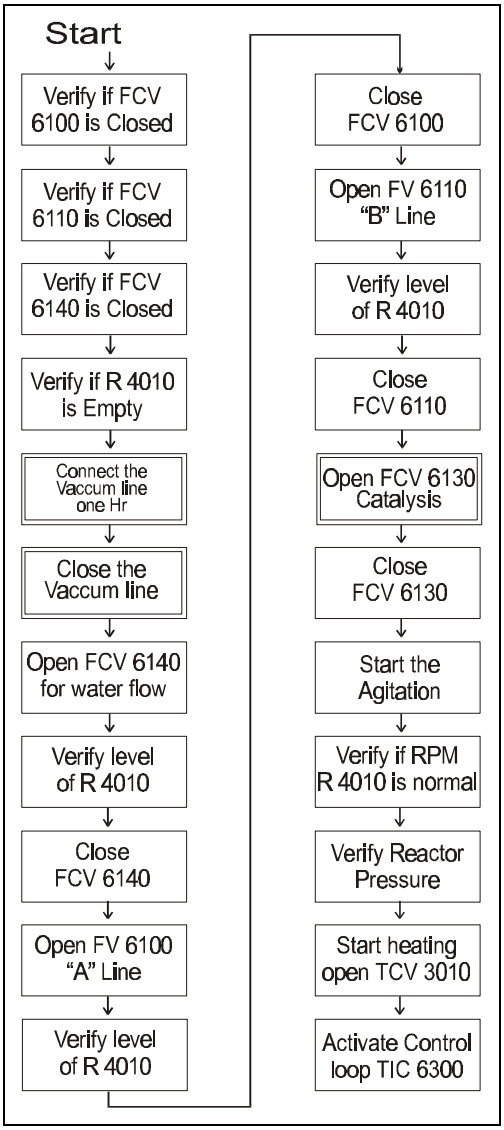



Fig. 7. Parcial event cascade for reactor start-up.

In real time, after the sequence of tasks pertaining to a block $I_i$ or $O_i$ is supervised, and the state of the process variables can be classified as normal, the system labels the block of initial conditions $I_i$ as normal, and begins with the supervision of the corresponding batch operation $B_i$. All the faults that cannot be identified in this step (for example due to the existence of not measured variables) must be included in the model (fault models) of the unit $B_i$.

*Models of single batch units. The Batch Reactor*

The reaction (highly exothermic) taking place in the reactor (R4010) is the following:

**A + B Û C + D**

The reaction takes place in liquid phase. A PID Control Structure is used to control the Reactor

Temperature. The reaction is self-accelerated by the produced heat. There are several design parameters and safety measures adopted at the design stage, like the cooling water flow system, the reactor design temperature, pressure valve relief, etc.

Figure 8 shows the optimal (normal) temperature evolution along a complete cycle (normal operation). Reactants are initially charged at environmental temperature (25°C). At this temperature, the reaction rate is negligible. Thus, all the mixture must be heated using the reactor jacket system. This heater is utilised to activate the reaction. It is operated with water at 60°C. After the reaction is fired, the reaction heat produces the mixture heating; thus, temperature grows suddenly. When a temperature greater than 60°C is achieved, the reactor jacket becomes a cooler.

Nevertheless, due to the high reaction rate, it is necessary to add an additional cooler to avoid the reactor run-away. The control system activates the cooler C100 when the mixture temperature achieves 90°C. C100 works with cold water (25°C). In this way, the reaction is abated, so the temperature achieves the maximum and starts decreasing.

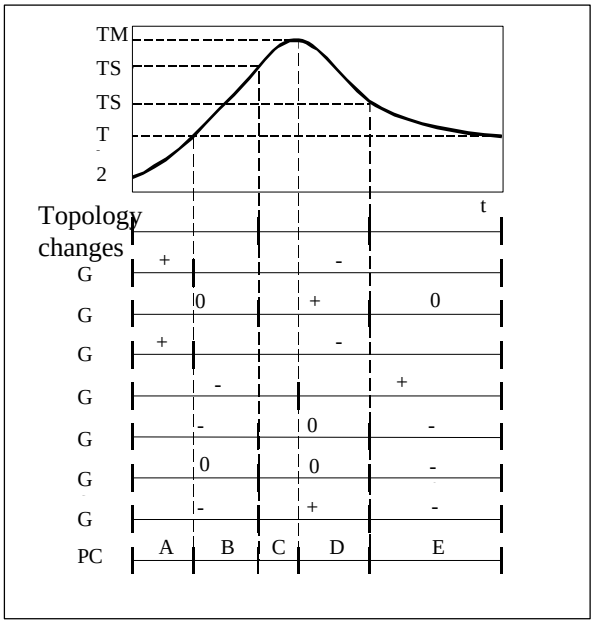



Fig. 8. Reactor Temperature Evolution

A dynamic simulator for the fault propagation estimation was implemented. The model consists of an ordinary differential equation system and an algebraic equation system originated from energy and mass balances, controller models, constitutive equations and physicochemical correlations.

For the PCBs determination, the first step is to analyse the topology changes, operative procedures, etc. We have five PCBs for this system (see Fig. 8). A mean number of 20 faults clusters were modelled and simulated for each PCB.

For the system performance evaluation, all the faults were simulated. Simulation outputs were affected by a Gaussian noise with several standard deviations and filtered by the detector module using an adaptive geometric filter ($\xi = 0.125$). In the detector module, a security band of ($2\sigma$) was used for the filtered signal.

Since we can write an adequate model for our reactor (an ODE system), we can use for fault modelling a "dynamic" SDG as explained in section II (see for more details Tarifa and Scenna, 1999, in which a FDS for a similar reactor was implemented using SDGs).

Here, we are more interested in the complete batch process, and how to relate the different sub-models among them (see Fig. 5).

The system presents an optimal performance considering different faults in cases where inputs include a noise level up to 15%. In those cases where the noise level is greater, the performance decreases, but can be considered satisfactory, taking into account that from the 100 faults tested (20 faults in each of the 5 PCBs), only 6 were exactly diagnosed but with a loss of resolution.

*The MBFDS for the Batch Distillation Equipment*

For implementing a MHFDS for the distillation column we use again the previously explained strategy. So, the first task is to model the Operative and Initial conditions blocks for the distillation column. It is known that we can divide the column operation in the start-up period, and the production period. The start-up procedure includes different stages. The first is the hydraulic period in which the holdup and vapour and liquid flowrates are not completely developed. This period is operated at total reflux. After this period, when the hydraulic profiles are developed, the reflux ratio is switched to the design value (according to the operating policy). In this stage, the different products are isolated by pumping them in the corresponding storage tanks. All these steps are indicated in the process recipe. So, the sequence of tasks must be processed as explained in the above section to obtain the rules representing the normal and abnormal states (operative and initial conditions blocks).

Indeed, we need to model the distillation process itself. The principal point is to identify the different PCBs. For the ternary column here analyzed, we have five PCBs, corresponding to the different operation modes and the topology of the system. More details for PCBs determination for a similar system can be found in Scenna *et al.* (2000).

For faults modelling, here we will use a set of Self-Organising Neural Networks (SONN) –each one for each PCB-. The self-organising model belongs to the class of vector coding algorithms, because the model provides a topological mapping that optimally places a fixed number of vectors into a higher dimensional input space. For the training process the Self-Organizing Mapping (SOM) algorithm is used. In a SOM structure, the neurons are placed at the nodes of a lattice that is usually one-or two-dimensional. The

neurons become selectively tuned to various input patterns (vectors) or classes of input patterns in the course of a competitive learning process. In competitive learning the output neurons of the network compete among themselves to be activated or fired, *with the result that only one output neuron or one neuron per group, is on at any time*. The location of the neurons so tuned (i.e. the winning neurons) tends to become ordered with respect to each other in such a way that a meaningful coordinate system for different input features is created over the lattice. SOM is therefore characterized by the formation of a topographic map of the input patterns, in which the spatial locations (i.e. coordinates) of the neurons in the lattice correspond to intrinsic features of the input patterns.

## 5. The MHFDS Performance

In the above points we have analysed each system individually. Here we must consider the complete system. Due to the time modularity, the diagnosis system can work efficiently supervising each block at once, according to the defined modular architecture. So, the efficiency of the system is directly associated to the efficiency of each module, plus the ES capability to handle the importation and exportation of information among sub-models. Up today, the MHFDS is implemented at a level of a prototype.

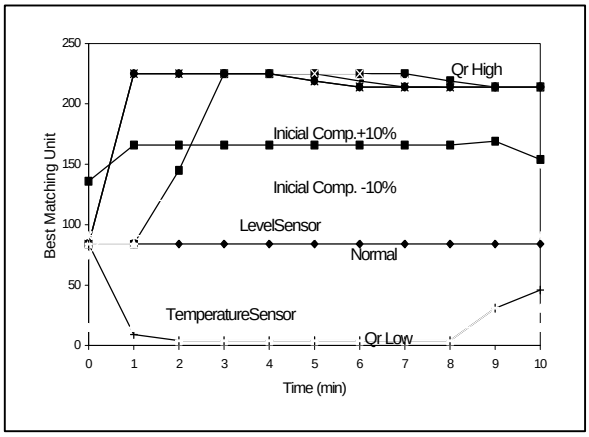



Fig. 9. SONN outputs

In Fig. 10 typical SONN outputs are shown for a set of faults pertaining to the domain of the first PCB of the distillation column. These temporal trajectories are processed by the ES to reach the final diagnostic. For cases in which process noises or outliers produce abrupt changes in the "reduced fault trajectories" (indicated in Fig. 9), we can use a supervised NN for processing the SONN outputs. So, in this alternative we are defining a hybrid approach processing the outputs of the SONN (a chain of symbols) as a grammatical interpretation problem. In future works, an architecture like the one indicated in Fig. 10 will be analysed.

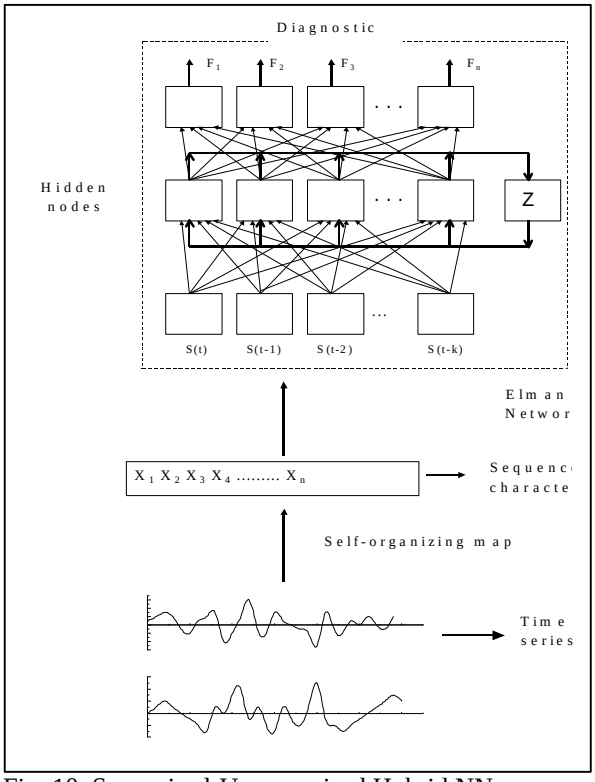



Fig. 10. Supervised-Unsupervised Hybrid NN.

## 6. Conclusions

In this work, a general framework for implementing a MHFDS and supervision toolkit for batch processes is presented. The objective is the problem presentation, to draw a general sketch for the problem solving task, and to show the capability of the proposed system by using a generic batch process as example.

It is concluded that for the achievement of a general scope and a satisfactory efficiency, a hybrid hierarchical system must be implemented. In fact, the temporal evolution of the process, the high non-linearity of the model and the different nature of the knowledge types to be represented, impose a multiple-modelling cooperative approach. Here, we use an ES working as a control level organising the real time reasoning process.

However, some points must be emphasised. For example, while the information about the initial conditions pertaining to the first blocks ($O_0$ and $I_0$ ) can be easily encoded, for all the subsequent temporal blocks we must analyse potential faults effects occurred in previous stages, which may not be detected in the active one due to insufficient measurements in real time. It is interesting to note that while for the adjacent stages this is easier, faults originated in some blocks will often present their symptoms in blocks far away along the time horizon of the process.

This phenomenon complicates the diagnosis task, because the linking among non-adjacent stages are more difficult. Indeed, a complex event tree can appear. This situation is more probable when a poor

instrumentation of the process exists. Thus, an exhaustive analysis of the propagation paths must be done.

It is also important to stress that the hybrid system here presented is specially suited for implementing supervisory or fault diagnosis systems devoted to highly discontinuous processes. In fact, start-up or shut-down operations are characterised by a set of actions (recipe) performed by an operator or by a computer. Indeed, for continuous or discontinuous processes, after a fault is detected, it is very difficult in real time to supervise the operator actions. In fact, these are events produced in real time, and directly affect the fault evolution. Of course, for implementing a software package for operator training we must consider this problem. Again, the methodology and the architecture here presented are a good starting point for implementing this tool for batch processes.

**References**

Jacobs, R. & M. Jordan, "Learning Procedure Control Strategies in a Modular Neural Networks Architecture", *IEEE Transactions on Systems, Man and Cybernetics*, **23**, 2, March/April (1993).

Kramer M. and B.L. Palowitch, "A rule-based approach to fault diagnosis with SDGs", *AIChE J.* **33** (7), 1067-1078 (1987).

Henley and Kumamoto, Reliability Engineering and Risk Assessment, Prentice Hall, USA (1981).

Himmelblau, O.M., Fault Detection and Diagnosis in Chemical and Petrochemical Process. Elsevier, Amstedam, (1978).

Iri M., K. Aoki, E. O'Shima and H. Matsuyama, "An algorithm for diagnosis of system failures in the chemical process", *Comput. Chem. Eng.*, **3**, 489-493 (1979).

Shamma, J. and M. Athans, "Analysis of gain Scheduled Control for non-linear plants", *IEEE Trans. Aut. Cont.*, **35**, 898-90 7(1990).

Scenna, N., S. Benz, B. Drozdowicz and E. Lamas, "A Diagnosis System for Fault Diagnosis in Batch Distillation Columns", ESCAPE10, *Computed Aided Process Engineering*, **8**, 805-810 (2000).

Scenna, N., "Some Aspects About Fault Diagnosis in Batch Processes, *Reliability Engng. & System Safety*, in press, 2000.

Tarifa, E. and N. Scenna, "A Methodology for Fault Diagnosis in Large Processes and an Application to a Multistage Flash Desalination Processes". Part I. Theoretical Considerations, *Reliability Engng & System Safety*, **60**, 31-41 (1998a).

Tarifa, E. and N. Scenna, "A Methodology for Fault Diagnosis in Large Processes and an Application to a Multistage Flash Desalination Processes". Part II. An application, *Reliability Engineering & System Safety*, **60**, 42-52 (1998b).

Tarifa, E. and N. Scenna, "A methodology for fault diagnosis in batch reactor", *Comput. Chem. Eng.*, S223-S226 (1999).

Umeda T., T. Kuriyama, E. O'Shima and H. Matsuyama, "A Graphical Approach to Cause and Effect Analysis of Chemical Processing Systems", *Chem. Eng. Sci.*, **35,** 2379-2388 (1980)

**Received August 30 1999;**
**accepted for publication May 29 2000**
**Recommended by Bandoni, Diaz and T. Pinto**