Expert Supervision Based on Cases

Joaquim Meléndez Joan Colomer Josep Lluis de la Rosa

eXiT Group
IIiA & LEA-SICA
Universitat de Girona
Av. Lluís Santaló s/n
E-17071, Girona
SPAIN

Abstract- **The paper is focused on taking advantage of big amounts of data that are systematically stored in plants (by means of acquisition and monitoring systems, i.e. SCADA systems), but not exploited enough in order to achieve supervisory goals (fault detection, diagnosis and reconfiguration). The methodology of case base reasoning (CBR) is proposed to perform supervisory tasks in industrial processes by reusing stored data. The goal is to take advantage of experiences, registered in a suitable structure as *cases*, avoiding the tedious task of knowledge acquisition and representation needed by other reasoning techniques as expert systems. An outlook of CBR terminology and basic concepts are presented. It is discussed how to adapt CBR in performing expert supervisory tasks taking into account the particularities and difficulties derived from dynamic systems. Special interest is focused in proposing a general *case* definition suitable for supervisory tasks. Finally this structure and the whole methodology is tested in a application example for monitoring a real drier chamber.**

1. INTRODUCTION

This paper describes how case based reasoning (CBR) methodology can be applied to improve expert supervision approach by exploiting data acquired from sensors. The goal is to structure historical data related to representative situations as *cases* in order to be used to diagnose future situations by analogy. *Cases* are data structures containing episodic knowledge of process behaviour during exemplar or interesting situations succeeded in the past.

Main difficulties in applying CBR in industrial process are because the dynamic behaviour and the necessity of taking into account temporal considerations in the definition of *cases* and during the reasoning procedure. The discussion presented in this paper takes into account the different typologies of process (continuous /batch /discrete-event driven) and the necessity of combining numerical and symbolic information in the supervisory tasks. A generic *case* definition is proposed to achieve supervisory goals in the domain of dynamic systems. It includes the definition of similarity functions according to time evolution for *case* retrieving. The implementation of the CBR methodology for supervision is proposed and tested within an illustrative example.

The following two sections present the concepts and terminology involved in both expert supervision and case based reasoning. The goal is to present both domains and make incidence on the benefits of using a case based methodology to achieve supervisory goals under a knowledge-based approach. The benefits of this approach are discussed in section 4 and 5, while section 6 is dedicated to point the main requirements in CBR. Some contributions, related to specific applications in the continuous domain are revised in section 7. Section 8 is centred to discuss about the problem of *case* representation. A general *case* definition, suitable for any process, is made in this section. The next one discusses about similarity criteria to be used for comparing and retrieving *cases*. Finally an example is proposed in order to show the benefits of using CBR to improve the control and supervision of a direr chamber.

2. EXPERT SUPERVISION

Supervision has grown as an active research topic for the last twenty years. It includes three basic stages named: fault detection, fault diagnosis and reconfiguration. There is not an exact limit between the two first tasks and multiple methods have been proposed to achieve them under certain restrictions. Two main streams can be distinguished in the research community according to the methodology used for this purpose.

The first approach takes advantage of knowledge about normal operation behaviour embedded in analytical models. The goal is to run the model in parallel with the real process and fitting the same inputs to both. Differences between both outputs (*residual*) are used to detect *discrepancies* (fault detection) of the real plant respect to the desired behaviour (normal operation represented by the model). These *discrepancies* are analysed in order to estimate fault size and find out signatures that allow faults to be located and identified (fault diagnosis). Main drawbacks in applying this methodology are because the difficulty in both, obtaining models accurate enough and describing suitable signatures.

The second point of view is focused on applying knowledge-based techniques (artificial intelligence) in order

0-7803-7241-7/01/$10.00 (c)2001 IEEE

to detect process behaviour which links effects with causes (origin of faults). Effects, also known as *symptoms* (from diagnosis point of view), are obtained from observed data and they are used to deduce the process diagnosis according to some knowledge previously acquired from past situations, experience and/or physical constrains. In this approach the limits between fault detection and diagnosis are not always clear and more emphasis is put on diagnosis. Different approaches could be distinguished according to the knowledge they use. We talk about *model-based diagnosis* when structural and functional knowledge about process is used to obtain qualitative models (normal operating and faulty) in order to propagate observations, from the real process, into such models with the purpose of validating diagnosis hypothesis. On the other hand, we talk about *expert supervision* when expertise and/or experience are codified (compiled) into reasoning tools. Typical tools used with this purpose are *expert systems*. The aim is to take advantage of expertise to decide and diagnose about process behaviour (normal or faulty operation) avoiding the construction of a model (the model is implicit in the knowledge base). Main drawbacks in such domain are because human knowledge is related to concepts and symbols whereas process acquisition systems provide monitoring systems with numerical data. Consequently, knowledge based decision systems are usually forced to work in a higher level of abstraction using symbolic variables instead of raw data coming from sensors. Numeric-to-symbolic interfaces have to be built and knowledge must be adapted to reason about these symbols [29][5].

Reconfiguration is the last step in a supervisory system. It follows to fault detection and diagnosis. It consists of proposing changes in the plant using the diagnosis result in order to keep the process under specifications when a faulty situation (non-catastrophic) is detected and diagnosed. Simple actions could be set points adjustment or tuning parameters in the controller. However, other actions related to maintenance or planning (control or system) could be proposed according to a predefined strategy (i.e. repair manuals, maintenance schedulers, planners and agendas).

3. THE CBR CONCEPT AND TERMINOLOGY

Case Based Reasoning (CBR) is a simple methodology proposed to problem solving by using previous experiences. It is based on Schank's dynamic memory models ([23]) and the basic idea is focused on the hypothesis that "similar problems have similar solutions" or -put in the other way round- "you can reuse the solution of a similar problem in order to solve your actual problem" ([26]). In order to achieve this purpose CBR methodology proposes a four-step cycle. It basically consists in **R**etaining experiences as *cases* for a further **R**euse. *Cases* are registers containing a description of a problem and its solution. The aim is to reuse these *cases* for solving new problems by analogy. In presence of a new problem, the basic procedure consists of **R**etrieving analogue *cases* and reusing their solutions. Reuse implies an adaptation procedure of the retrieved solutions that is finished with a **R**evision. After validation, the cycle is completed by Retaining the solved situation (problem + new solution). These operations are known as the 4R of the CBR cycle (See Fig. 1). In practice it is often confused the separation between reuse and revise and it is common to merge both in one operation named adaptation.

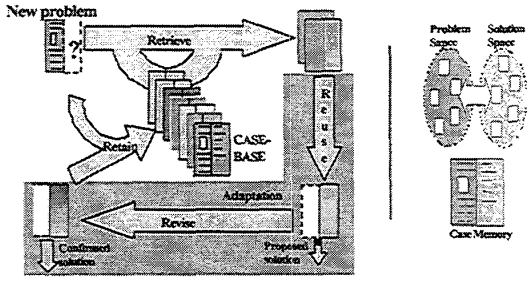

Fig. 1 *Cases* and CBR cycle: Retrieve, Reuse, Retain and Revise.

Cases are stored according to a suitable structure, named case memory, into case bases where indexing mechanisms are used for an efficient retrieval. CBR foundations and a detailed description of this methodology based on the **4R** are wider described in [1] and [9].

4. USING CBR FOR DATA EXPLOITING IN THE INDUSTRY

SCADA (Supervisory Control And Data Acquisition) software is the most extended software used for monitoring in the actual industry. Despite the meaning of letters in the acronym, supervision is a poor feature (according to description done in the previous section 0) of this type of software. It is basically reduced to simple mechanisms for alarm generation based on thresholds (absolute and/or relative) applied to signals or rates of change (ROC). On the other hand they incorporate some important facilities to store acquired data usually linked to standard data bases (DBs). It means that large amounts of data (parameters, measurements, controlled variables, alarms, events, etc.) are captured, dated and stored on such DBs. All the information about process could be registered but the real challenge is how to take advantage of them for improving manufacturing.

This growing amount of data, systematically stored and poorly used, represents a certain view of the reality being dealt with (the process behaviour). Another view to this part of the real world is captured by the experiences that people gather as part of their daily information handling and problem solving effort ([9]). Consequently, CBR is intended to be a natural extension of actual monitoring systems with the aim of extending its capabilities by merging operators knowledge with stored data. The aim is not to substitute the experienced

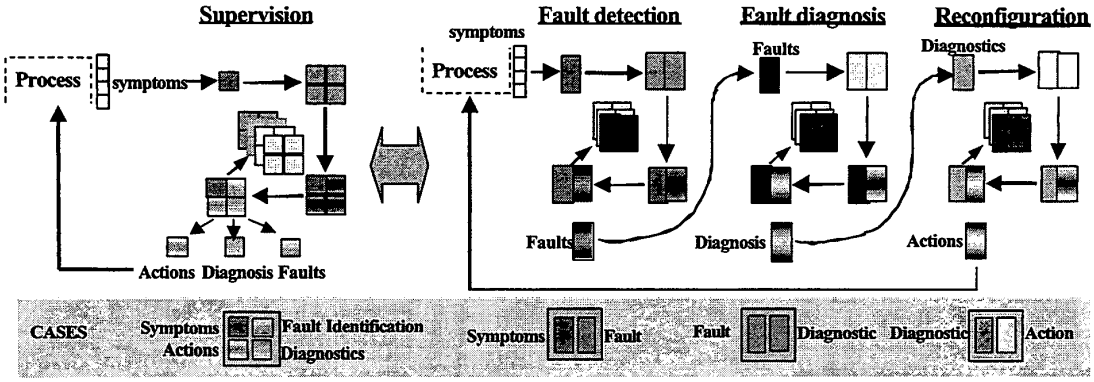

Fig. 2 CBR cycle applied to supervisory tasks.

operator but to assist them in their activities taking benefit of an efficient use of stored data.

DBs are efficient means of storing and retrieving large volumes of data widely used in the industry. Then, what's the difference between DB and CBR? The basic difference between DBs technology and CBR is in the retrieval concept. The retrieve procedure in DBs basically consists in an exact, or partial (using wild characters, i.e. '*'), matching to queries. Although, it is also allowed to use other operators (as ranges, comparison or thresholds) the retrieved information always have a perfect matching with the specified conditions. Instead of this, CBR defines the retrieve procedure in terms of distance. It means that a similarity (distance) criteria, or alternative method (i.e., inductive retrieval), must be used to retrieve *cases* instead of a perfect matching of conditions (distance equal zero). However, data base mechanisms (i.e. SQL queries) can be improved with explicit knowledge of the relationship between concepts in a problem domain in order to measure similarity ([25]). Another important difference in the actual CBR systems points to the incorporation of general knowledge about the particular problem to be solved. This knowledge is used to define the case base structure and general relationships among attributes contained in *cases*. Domain knowledge is used in order to improve case base management.

5. CBR METHODOLOGY AND SUPERVISION

It is possible to apply CBR methodology in the three stages of supervision (fault detection, diagnosis and reconfiguration) avoiding model building. According to expert-supervision point of view, the supervision problem can be treated as associations among *symptoms*, represented by acquired data from processes in faulty situations, their diagnosis and the proposed actions. Thus, *cases* have to contain passed experiences represented by associations among *symptoms*, faults, diagnostics and actions, acquired in specific situations. Then, the identification of a *symptom* will be used for retrieving associated diagnostics and actions in order to overcome it.

The implementation of this approach could be performed according to different strategies. Fig. 2 shows two possible examples. In the first one, on the left, *cases* are though to contain the whole information related to these experiences of supervision while the second, on the right, information is spread into three separated *cases* (faults, diagnosis and actions). In the last approach the CBR cycle would be applied three times to recover an action from a given symptom. Moreover, according to case structure, CBR cycle can be applied to retrieve partial information using different inputs in the retrieval. It means that the same case base could be used in fault detection or as a consulting assistant to obtain information about *symptoms* related to a specific fault.

6. KNOWLEDGE REQUIREMENTS IN CBR

CBR avoids having to encode the knowledge that would be required to automate the first principle reasoning. Instead of this, the problem solving knowledge must be present in higher stages. Richter [16] describes four containers in which CBR stores knowledge. These are directly related with the implementation of the supervisory system using the CBR approach:

1) *Vocabulary knowledge*: used in the description of the whole domain. It is application dependent and defines the environment of the problem to be solved.

2) *Cases knowledge*: stored in the case base as a set of structured experiences represented by *cases*. The definition of a suitable structure (case memory) is needed in order to facilitate retrieval procedure. A (or various) generic structure for a case must be defined but also how they are linked (indexed) in the case base.

3) *Retrieval knowledge:* used in the definition of retrieval algorithm in order to find similar *cases*. It is necessary to define how two *cases* can be compared, which attributes are the most relevant and the limits for considering two *cases* similar, or analogue, enough.

4) *Adaptation knowledge*: necessary for a suitable reuse of the retrieved solutions. Differences between actual case

and retrieved *cases* must be taken into account in order to reuse the retrieved solution.

Major efforts in building a supervisor based on CBR must be centred in obtaining the best definition for these containers according to the supervisory goal and the process restrictions. The following sections are centred in the definition of a generic case structure suitable for supervisory goals.

7. AN OUTLOOK OF CBR APPLICATIONS INVOLVING TIME VARYING DATA

Contributions of CBR (Case Based Reasoning) in the domain of time varying data are quite recent and they are reduced to applications in some specific domains. Some contributions have been done in mobile robotics (for example in prevention of collisions [20] or for adaptive planning [18]), weather prediction ([7] and [15]) or process control (of waste water plants [17], power systems [11], discrete plants [3] or batch processes [12]).

Control and supervision of process is a domain where CBR can rapidly extend its benefits because data is systematically collected and registered for its further analysis. However, CBR has not been extensively applied in this domain because the difficulty of working with temporal representations of behaviours. In fact, most of applications involving CBR in the industry apply this methodology as a complementary technique for knowledge acquisition. For instance, in [3] CBR is proposed as conceptual framework in which to store operator experience and later provide that experience to other operators to facilitate situation assessment and solution formulation. So, the use of CBR is reduced to assist control plant planning. In a similar, way CBR has been used in the manufacture industry for planning, diagnosis, troubleshooting, maintenance, quality management and so on [4]. A similar approach is performed to test analogue circuits by using dictionaries of faults ([2],[6] and [14]). All of them are application domains where reasoning is performed off line and adaptation of retrieve *cases* does not exist. It means that this reasoning does not affect process dynamics and CBR has been used neither for control nor in supervision.

8. CASE REPRESENTATION FOR DYNAMIC SYSTEMS

Reasoning about dynamic systems is not an easy task. Moreover, complexity increases when merging human perceptions (about process behaviour) and evolution of physical variables (provided by sensors) is needed. Numerical methods are needed to deal with acquired data from sensors and, on the other hand, operators use basic process knowledge and accumulated experience for situation assessment and propose actions. They are two different views of the same reality ([9]), the process, that must be combined in order to improve the global knowledge of process needed in process supervision.

Many efforts of artificial intelligence community have been oriented to integrate both numerical (or model based) and knowledge based methods (e.g. [29], [30], [31]) for supervision when models are not available. Difficulties in automating this reasoning appears when numerical information must be interpreted (feature interpretation) in order to be supplied to knowledge based systems. So, CBR methodology points a guideline based on the representation of meaningful sequences of acquired and abstracted data as *symptoms* and its further use to identify situations.

CBR difficulties start in the definition of *cases*. *Cases* are inherently static registers, but significant information about process behaviour is due to changes and transitions. Main source of information is provided by signal evolution. Process dynamics depends on the interaction of its components and the materials involved. Faults are variations of these interactions that will be present in the measurements. The richer the signal (in the frequency contents sense) the greater the amount of information. A constant value in a signal, through time, gives a poor information about process behaviour (only steady state and its value). Therefore, reasoning about process dynamics implies to store historical evolution of variables in *cases* in a suitable form. The register of previous situations has to contain history of such situations represented by variables. This leads to the continuous problem of CBR pointed, and still open, by Ram and Santamaria ([19],[18]) related to *case* representation: "How should 'continuous *cases*' be represented?, When do *cases* start and end (in the temporal sense)?, When are two experiences different enough to warrant consideration as independent *cases*? What is the scope of a single *case* ?".

A Some antecedents

Several approaches have been proposed for *case* definition involving dynamic process and most of them are clearly application dependent. For instance a particular representation of *cases* is proposed in [24] for modelling dynamic systems. It is used to obtain a mapping between input (multiple input) and output (single). It includes derivatives in the case definition as a sensibility relationship. The model obtained is then used for fault detection as in a model-based approach.

On the other hand the majority of applications involving CBR approach under dynamic systems define *cases,* C, simply as process states in a time instant (i.e. $C=X(t_i)$. It means that *cases* have been defined to represent the values of a set of process variables in an instant (as in [18]) or an average of values in a predefined period of time (as in [17]). When using this simplification, process history is not taken into account, and consequently its dynamics is poorly represented. Thus, *cases* are used as static pictures of process behaviour. Additional drawback for this representation is because states are not always well defined and their number could be infinite or indeterminate. In fact, the size and structure of *cases* needed to represent extended

experiences clearly depends on the application. It has been an open issue from the beginning of continuous CBR ([8]). Moreover, dynamics, i.e. transients, can only be represented by means of sequences of data.

Although, state identification is important, it is not always possible and major information is obtained from sequences of data and state transitions. On the other hand, it is not necessary to store the whole state evolution for reasoning about it. According to these considerations a better representation, suitable for diagnosis of dynamic systems under a CBR methodology, is suggested in [27]. In this approach, named Dynamic CBR, measures are mapped into a features space defined by qualitative indices (static and dynamic). Dynamic features are represented in an interval by instantaneous variations and qualitative trends.

Based on this representation, a simple improvement is proposed: The case definition (C) is expanded to a non fix temporal interval (t_o,t_f). This interval starts and finishes with significant events. Information in these instants is stored (C_o and C_f). Process behaviour between both events is then represented by the evolution of the process variables (measures and actions) vector, X(t), in a suitable form (abstraction) for representing *symptoms* related to faults (or normal behaviour). See Fig. 3.

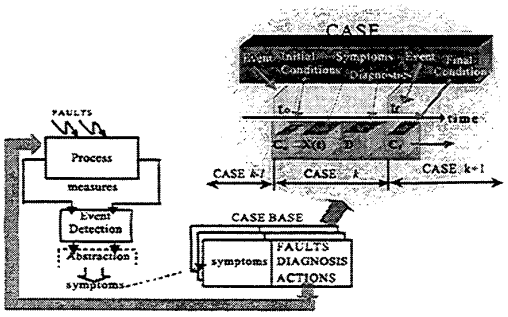

Fig. 3 Case generation from event detection and symptoms description.

The case structure definition is completed with slots describing faulty situation, diagnosis and actions taken in presence of the stored symptom. Any contextual information useful for monitoring tasks (e.g. explanation of performed actions, advises, or commentaries of normal operating conditions) could be added in this field (D).

Using initial (C_o) and final (C_f) conditions (related to system state when events are given) in the case definition, the general expression describing a case is given by the following expression:

$$C = \langle (C_o, t_o), X(t), (C_f, t_f), D \rangle \qquad t_o < t < t_f \qquad (1)$$

This formulation is compatible with other proposals described previously for planning and diagnosis. The vector X(t) is thought to represent meaningful information related to process in this interval of time (*symptoms*). A simple approach implies to register raw data (acquired sequences from sensors) and to define C_o and C_f as the initial and final values of this vector ($X(t_o)$ and $X(t_f)$). In a more elaborated representation more abstract and significant information could be used. For this proposal, statistical parameters, estimation methods or others that offer numeric to symbolic conversion can be applied [5]. The extracted (or abstracted) information is then used to characterise dynamics in this interval of time in a compact representation related to expert observations.

In [28], *cases* are built from acquired data by extracting features and estimating process operation conditions for controlling the charge in order to keep a furnace stable. Extracted features are both temporal (moving average, integral value, fluctuation, etc.) and spatial (radius direction, tangential direction, etc.) related to burden and gas flow distribution.

B Continuous systems

When system response is near to linear systems behaviour and the order and the state vector are available (measurable or estimated) the best strategy for supervision is based on the use of state vector under a model based approach. The state vector contains the whole information needed to define the system behaviour. A case representation for them would coincide with the state vector (Considering X(t) as the state vector: C=X(t)). See Fig. 4.

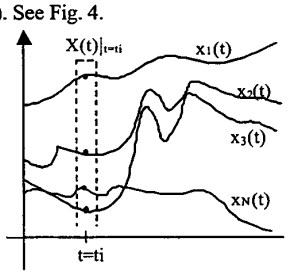

Fig. 4 State vector in a N order system

On the other hand, when dynamics is difficult to be modelled, the order is unknown, the systems has non linear behaviour or it changes from time to time; then, CBR approach is proposed for supervising complex dynamic systems without using models. In such situations the state vector is not available, and *cases* has to be built, according to previous definition, by adding signals history in order to compensate the uncertainty related to dimension of the state vector. The use of temporal series of acquired signals, instead of a single vector in one time instant improve the characterisation of process behaviour (Fig. 5).

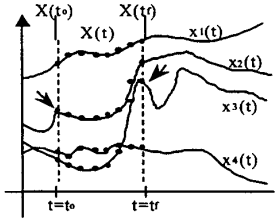

Fig. 5 Case representation based on temporal series of an N order vector.

$$C = \langle (X_o, t_o), X(t), (X_f, t_f) \rangle \qquad t_o < t < t_f \qquad (2)$$

The length of a *case* is not fixed and it depends on the presence of two consecutive events. Although, these events are represented by maxima in $x_2(t_o)$ and $x_3(t_f)$ in the Fig. 5, they could be obtained from other mechanisms. The goal is to register system behaviour when something significant happens. In the same way, the type of information represented is not restricted to sampled signals between both instants but qualitative or abstracted information could also be used. This is represented in next Fig. 6 where tendency of signals is represented by a straight line (tendency) and presence of oscillations by a curved line. Thus, in this example $X_a(t)$ represents the abstracted information, not the acquired samples of signals. A symbolic representation of this vector could be the following:

$$x_{a1}(t) = (\uparrow, t_{10}, t_{11}), (—, t_{11}, t_{12}), (\uparrow, t_{12}, t_{13}) \qquad (3)$$

$$x_{a2}(t) = (\downarrow, t_{20}, t_{21}), (\Uparrow, t_{21}, t_{22}) \qquad (4)$$

$$x_{a3}(t) = (\downarrow, t_{30}, t_{31}), (\Uparrow, t_{31}, t_{32}) \qquad (5)$$

$$x_{a4}(t) = (\uparrow\downarrow\uparrow, t_{30}, t_{31}) \qquad (6)$$

This type of representation is also adequate for batch systems where specific events are defined by the execution of recipes. In such systems process variations (reconfiguration, set points, etc.), performed according to the recipe, can be used in the case definition as significant events.

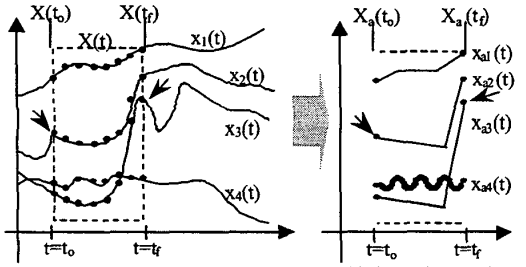

Fig. 6 Case representation based on abstracted information obtained from variables acquired between two consecutive events.

C *Discrete-event systems*

Discrete-event systems are driven according to changes experimented by the monitored variables. Such changes are defined as events, in the sense explained in the previous section. An event is interpreted as a change in the state of the system and the complete operation is performed as a sequence of states. Thus, *cases* can be build using the same structure presented before but taking into account the different nature of acquired variables. Significant events are used to start and finish a *case* and process history between them is registered as a dated sequence of events happened to the whole set of monitored variables. Next figures depicts this representation for a case started in t_o when x_3:1→0 and finished in t_f when x_2:0→1. The temporal axis shows the instants in which events take place.

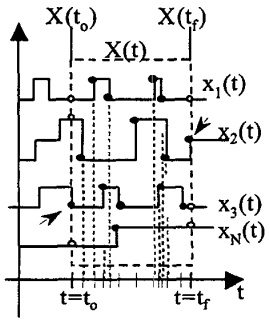

Fig. 7 Case representation in a discrete-event system.

The representation of *case* in Fig. 7 according to this expression:

$$C = \langle (X_o, t_o), X(t), (X_f, t_f) \rangle = \\ = \langle X(t_o), X(t), X(t_f) \rangle \qquad t_o < t < t_f \qquad (7)$$

is given by:

$$X(t_o) = (x_1(t_o), x_2(t_o), x_3(t_o), \ldots, x_N(t_o)) = (0, 2, 0, \ldots, 0) \qquad (8)$$

$$X(t_f) = (x_1(t_f), x_2(t_f), x_3(t_f), \ldots, x_N(t_f)) = (0, 1, 0, \ldots, 1) \qquad (9)$$

And process history between instants t_o and t_f is given by the dated sequence of dated events (represented by the couple (event, t_e)) appeared in each signal:

$$X(t) = (x_1(t), x_2(t), x_3(t), \ldots, x_N(t_o)) \qquad (10)$$

In the example these vectors take the following values:

$$x_1(t) = ((1,t_{11}), (0,t_{12}), (1,t_{13}), (0,t_{14})) \qquad (11)$$

$$x_2(t) = ((0,t_{21}), (2,t_{22}), (1,t_{23})) \qquad (12)$$

$$x_3(t) = ((1,t_{31}), (0,t_{32}), (1,t_{33}), (0,t_{34})) \qquad (13)$$

$$x_N(t) = (1,t_{N1}) \qquad (14)$$

The first couple in each vector corresponds to the first change detected in each signal from the initial state and the value in last one coincides with the final state stored in $X(t_f)$. This definition has been used in a laboratory plant in [13].

9. SIMILARITY FOR CASE RETRIEVAL

The most commonly used criteria for case retrieval are based on the concept of distance. They are used to obtain the k-nearest neighbours of a case, C^A, from a case base containing *cases*, designed by C^B. The following is a general expression used for distance calculation between *cases*. They are supposed to be composed by a set of N attributes x_i which similarity is measured by a function, *sim()*:

$$sim(C^A, C^B) = \left[\sum_{1}^{N} f(w_i) \cdot sim(x_i^A, x_i^B) \right]^{\frac{1}{r}} \qquad (15)$$

A common expression suitable for similarity calculation between attributes (*sim()*) is given by the following equations and r is a parameter that depends on it. It includes the Manhattan, Euclidean and cubic distances for numeric attributes and the overlap distance for symbolic:

1) Symbolic attributes[1] (r=1, overlap distance):

$$sim(x_i^A, x_i^B) = \begin{cases} 0 & x_i^A = x_i^B \\ 1 & x_i^A \neq x_i^B \end{cases} \quad (16)$$

2) Numeric attributes[1]:

$$sim(x_i^A, x_i^B) = \left| x_i^A - x_i^B \right|^r \quad \begin{cases} r = 1 \rightarrow Manhattan \\ r = 2 \rightarrow Euclidean \\ r = 3 \rightarrow Cubic \end{cases} \quad (17)$$

The importance of each attribute is usually weighted, w_i, in order to obtain the global distance. The election of these weights is performed according to the importance of each attribute with respect to the others in the *case*. Its influence in the distance criteria can be applied directly with a multiplier effect or using a function (normalised) of them $f(w_i)$. For instance, an exponential function could be used to emphasise differences among them as is suggested in [22]. This normalised and exponential weight-sensitive distance function based on Manhattan distance is used in [21] for a diagnostic application with this goal.

The use of distance-functions allows to obtain an ordered list of retrieved *cases* from the case base according to a similarity criteria. Thus, the most, in fact the K most, similar *cases* (K-Nearest Neighbour), can be used to adapt the new solution according to a transformational relation.

10. SUPERVISORY CONTROL OF A DRIER CHAMBER

A Process and operation description

Previous definition of *cases* has been used for supervising a cured meat drier [12]. This type of process are basically controlled by applying recipes according to composition of raw material, initial conditions measured before entered into the chamber and the desired output product. Recipes are sequences of set point changes in the controlled variables, i.e. temperature or humidity, and activation/deactivation of devices for specific purposes (i.e., compressors, ventilation fans and so on). The adequate sequence of actions is tuned according to experience and observations of process made by operators. Recipes are not strictly executed, since changes are introduced on line according to these subjective observations. Commonly, these observations are related to non measurable variables (smell, taste, feel, sense of touch etc.) that are very important in order to decide about final product quality, e.g. cured or mature degree, appearance, etc. As a consequence of these observations control strategies proposed by recipes are usually re-tuned by operators according to their experience and the particularities of the product being processed. This control strategy is often used in batch process in the food industry (dryers, ovens or reactors) is represented in Fig. 8 in a-two-variables simplified example. The variable named x_2 is used for control while x_1 is only used for monitoring. Process starts at X_o and the goal is to reach X_f under certain restrictions in the trajectory (see dotted envelopes in the figure below). For this proposal a set point recipe is proposed (dotted step-shaped line in x_2) but in a certain instant (t_1 and t_2) the operator decides to force a set point change (solid step-shaped line) according to the observations and his experience related to similar behaviour.

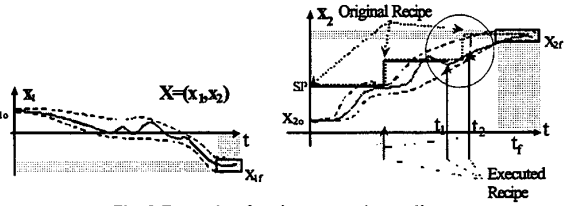

Fig. 8 Example of recipes re-tuning online.

The type of control is quite usual in the alimentary industry and those processes where not all the interesting variables can be measured. The goal is to automate the re-tuning procedure executed by operators taking advantage the situations previously stored according to the CBR methodology.

B Case Base structure

According to previous definition of *cases* (every case starts and ends with a significant event), an executed recipe could be understood as a sequence of *cases* starting and ending with a set point change, important observations, perturbations, alarms and so on. In this process, the goal is to achieve some desired conditions with some restrictions (duration and range of variables). We call a *batch* the complete execution of a recipe. In order to store the whole executed recipe significant events are used to define *cases* in a way that a batch consist of an ordered sequence of *cases* as is depicted in the Fig. 9.

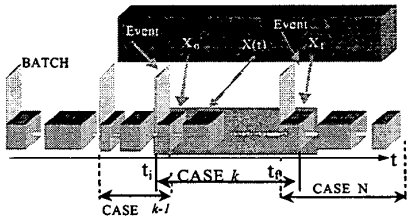

Fig. 9 A batch is an ordered sequence of *cases*.

It is important to notice that any *case* in a batch are chained in the way that the final conditions are the initial conditions of the following one (except the last one), and it is

[1] The indices A and B in the notation can be interpreted as Actual *case* and any *case* form the *case* Base.

dated when an event is done. The case base is a collection of executed batches and traced recipes composed by *cases*. It is organised in a way that *cases* could be managed independently although they are linked to the previous and posterior ones. Fig. 10 represents the case base and Fig. 11 shows how *cases* can be treated as independent entities.

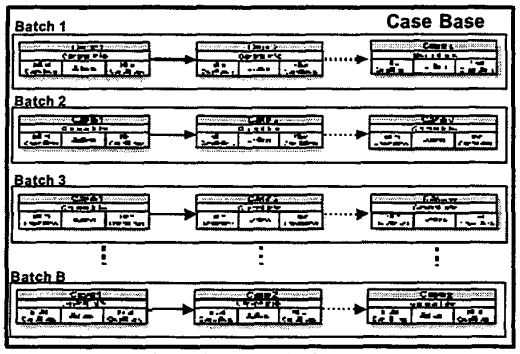

Fig. 10 Structure of a Case Base.

C *Retrieval procedure*

In this application the retrieval procedure firstly focuses the search on the batches by means of search trees using the most similar general conditions (initial conditions and final desired ones). This provides a partition of the case base where to deepen the search by matching events (initial conditions of *cases,* Xo). Then the distance between two *cases* defined by N variables is calculated according to a weighted function of them. Similarity function can take into account initial conditions, final conditions and transients. Then, three subsets of weights are used in the similarity function: for the initial conditions in the case (w_o), for the final conditions (w_f) and for comparing attributes related to the transient (w_t) between initial (t_i) and final instants (t_i) in the case.

$$sim(case^A, case^B) = \sum_{i=1}^{N} \left| Xo_i^A - Xo_i^B \right| \times Wo_i + \\ + \sum_{i=1}^{N} \left| X^A(t_i, t_f) - X^B(t_i, t_f) \right| \times Wt_i + \\ + \sum_{i=1}^{N} \left| Xf_i^A - Xf_i^B \right| \times Wf_i \qquad (18)$$

Weights are tuned according to the importance of each variable. It has been considered the uses of three sets of weights in order to take into account the temporal dependence of variables by using different weight at the beginning or at the end of the case. The previous expression takes into account the fact that some abstraction or manipulation could be performed to the variables in order to obtain more significant information represented in the *cases*.

When using only initial conditions (w_f=0, w_t=0) in the case for retrieval, the actual state is being compared with the beginning of stored cases. Thus, the retrieved episodes can be useful for short-term prediction (final conditions of this episode) and also to perform actions in possible faulty situations. On the other hand, the use of the three subsets of weights allows a better matching when the goal is to reuse the sequence of *cases*, belonging to a previous batch, linked to the retrieve one. This will be useful for long term prediction and planning (See Fig.12).

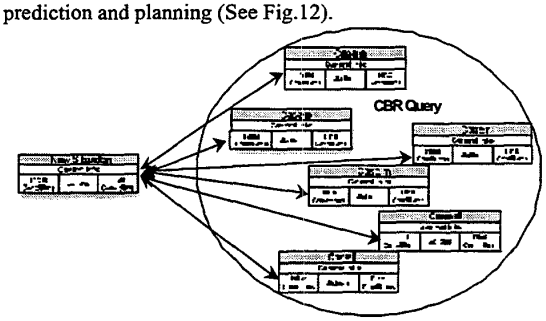

Fig. 11 Nearest Neighbour Case Retrieval

The goal is to reuse *cases* when similar initial conditions are presented in order to achieve a final product conditions (quality, time, etc.). This is the simplest way of reusing recipes but the proposal extends the applicability to a continuous improving of production. It means that comparison is performed continuously while production is running. In case of actual conditions were similar to the initial conditions of a registered episode (and also the desired goal, final conditions) actions performed in it could be reused to redefine the actual recipe (See Fig.12)

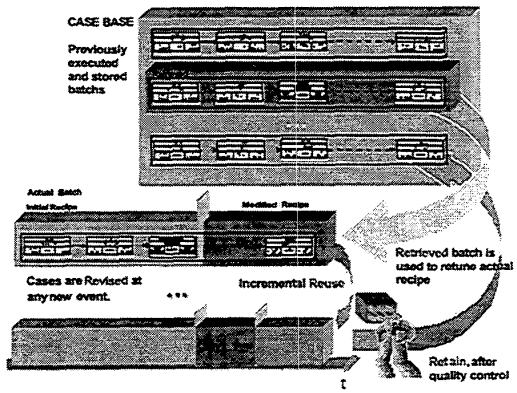

Fig. 12 CBR cycle for *recipes* re-tuning.

The process is supervised by means of a continuous monitoring and recipe improving according to CBR methodology. Initial recipes were proposed by operator experience (initial case base), but it is increased and modified according to new productions in the chamber. The CBR cycle depicted in the Fig. 12 runs according to the following procedure: When a production is started, an initial recipe for the whole execution, is proposed according to initial conditions and raw material composition. While product is being cured, some variables (chamber temperature and humidity relative, superficial humidity and temperature in a sample sausage, etc.) are acquired by the monitoring system,

or introduced by the operator according to observations. They are used in order to monitor the batch execution. This is performed comparing actual data with registered *symptoms* (K-nearest neighbour) in the previous stored executions. When deviations from desired constraints or a better execution are matched, then recipe is re-tuned by using the retrieved ones. Finally, at the end of production, quality is evaluated and the executed recipe is stored for a further reuse.

D Industrial Implementation

This supervisor has been used in pilot chamber. Basic architecture implemented is depicted in Fig. 13: The SCADA system (RSView from Rockwell) provides information (from process and CBR) to the user allowing visualisation of data and parameters modification. The user interface also allows to interact with the CBR application. In fact, it is used to revise and build cases involved in the CBR mechanisms.

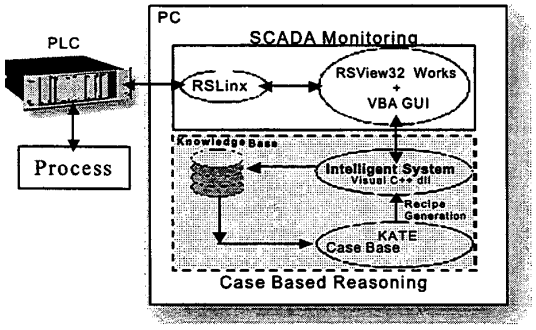

Fig. 13 Industrial Implementation

The case structured has been implemented using object technology by means of using Visual C++ and MFC facilities while the CBR inference is powered by KATE suite (Acknosoft). Predefined structure of cases in KATE obligates to build a dynamic link between both in order to preserve the desired case structure.

This system is now being tested in a pilot drying chamber in order to define the adequate dimension of the case base for an optimum use. Main goal is to test and improve this application in order to transfer research results.

11. CONCLUSIONS

CBR methodology has been proposed in order to improve actual monitoring systems. The complexity of working with time dependent systems is discussed and an overview of different applications involving CBR in the control and supervision domain has been done in order to propose a suitable case structure useful for any supervisory application independently of the variables nature. The goal underlying in this work is to take profit of acquisition systems from industrial processes. They are powerful sources of raw information that are not profited enough. An example of applying CBR methodology in a batch has been presented.

12. ACKNOWLEDGEMENTS

This work is partially supported by the projects ***"Advanced Decision Support Systems for Chemical/Petrochemical manufacturing processes – CHEM" within the (EC IST program: IST-2000-61200)*** and ***CICYT DPI 2000-0658 (Diseño de Agentes Físcos dinámicos. Aplicaciones Futuras)*** within the CICYT program from the Spanish government.

13. REFERENCES

[1] Aamodt A. And Plaza, E., "*Case*-based Reasoning : Foundational Issues, Methodological Variations, and system approaches", Artificial Intelligence Communications, IOS Press, Vol : 1, 1994, pp: 39-59.

[2] Bandler J.W., Salama A.E., "Fault Diagnosis of Analog Circuits", Proceedings of the IEEE , 1985, pp.1279-1325.

[3] Brann D.M, Thurman D.A, Mitchell C.M, "*Case*-Based Reasoning as a Methodology for Accumulating Human Expertise for Discrete System Control", Proc. of the IEEE International Conference on Systems, Man, and Cybernetics, October 1995, Vancouver, B.C., Canada, pp. 4219-4223.

[4] Britanik J. and Marefat. M, "*Case*-Based Manufacturing Process Planning with integrated support for knowledge Sharing", IEEE International Symposium on Assembly Task Planning, 1995, pp: 107-112.

[5] Colomer J., "Representació qualitative síncrona de senyals per la supervisó de sistemes dinàmics", 1998, Thesis Universitat de Girona.

[6] Duhamel P., Rault J.-C-, "Automatic Test Generation Techniques for Analog Circuits and Systems: A Review" , 1979, pp. 411-440.

[7] Hansen, B. and Riordan, D., "Fuzzy *case*-based prediction of ceiling and visibility", First Conference on Artificial Intelligence, American Meteorological Society, 1998, pp: 118-123.

[8] Kolodner J.L, "*Case* Based Reasoning", Morgan Kaufmann, Los Angeles, CA, 1993.

[9] Langseth H., Aamodt. A., winnem O.M., "Learning Retrieval Knowledge from data", IJCAI'99 Workshop ML-5: Automating the construction of *Case*-Based Reasoners, Anand, Aamodt and Aha Eds., Stockholm 1999, pp:77-82.

[10] Lenz M et al. "*Case*-Based Reasoning Technology. From Foundations to Applications", LNAI State-of-the-Art Survey, Springer, 1998.

[11] Long D, King R.L., Luck R., "Control of Power Systems Using *Case*-Based Reasoning", in 27 th Sotheastern Symposium on System Theory, 1995 , pp: 73-77.

[12] Meléndez J., De la Rosa J.L., Macaya D., Colomer J., "*Case* Based Approach for generation of recipes in

Batch Process Control", Butlletí de l'ACIA , num.22, octubre 2000, pp: 250-254.

[13] Meléndez J. Colomer J, Macaya. D. "*Case* Based Reasoning methodology for supervision", Accepted in the european control conference (ECC'01), 4-7 Sept, Oporto, (Portugal). 2001

[14] Milor L.S., "A Tutorial Introduction To Research on Analog and Mixed-Signal Circuits Testing", 1998, pp: 1389-1407.

[15] Nakhaeizadeh, G, "Learning Prediction of Time Series. A Theoretical and Empirical comparison of CBR with some other approaches", In, Proceedings of EWCBR'93, Richter, M.M., et al. (Eds.), Lecture Notes in Computer Science.VOL. 837, Springer-Verlag, 1993.

[16] Richter. M.M, "The knowledge contained in similarity measures". Invited talk on the ICCBR-95. http://wwwagr.informatik.uni kl.de/~lsa/CBR/Richtericcbr95remarks.html.

[17] R.-Roda I., Sànchez-Marrè M., Comas J., Cortés U., Lafuente J. & Poch M. "A Multi-paradigm Decision Support System to improve wastewater treatment plant operation", Workshop on Environmental Decision Support Systems and Artificial Intelligence, Technical Report WS-99-07, AAAI Press, Orlando (USA), July 1999 pp.68-73,.

[18] Ram A. , Santamaria J.C , "Continuous *Case*-Based Reasoning", Artificial Intelligence, (90) 1-2, 1997,pp: 25-77.

[19] Ram A. and Santamaría J.C, "Continuous *Case*-Based Reasoning" in the AAAI-93 Workshop on *Case*-Based Reasoning, 1993, pp. 86-93.

[20] Rosenstein M.T. and Cohen P.R., "Concepts From Time Series", Fifteenth National Conference on Artificial Intelligence, Madison, WI July 29, 1998.

[21] Sànchez M., Cortés, U., Lafuente, J., R.-Roda, I. and Poch, M. (1996). DAI-DEPUR: a Distributed Architecture for Wastewater Treatment Plants Supervision. Artificial Intelligence in Engineering, 10(3), pp: 275-285.

[22] Sànchez M., Cortés U., R-Roda. I., Poch M., "L'eixample distance: a new similarity measure for *case* retrieval", Proc of CCIA'98 (First Catalan Conference on Artificial Intelligence), Tarragona, Catalonia, 1998 pp: 246-253.

[23] Schank R.C., "Dynamic Memory. A theory of reminding and learning in computers and people. Cambridge University Press, Cambridge 1982.

[24] Tsutsui, H.; Kurosaki, A.; Sato, T.; Hiraide, Y., "Fault detection using topological *case* based modelimg and its application to chiller performance deterioration", Proceedings of the IEEE Instrumentation and Measurement Technology Conference, IMTC/94, vol.1 ,1994, pp: 390-393.

[25] Watson I., "Is CBR a Technology or a methodology?" in LNAN 1416, vol.2, Springer, 1998, pp:525-534.

[26] Wilke W., Bergmann R., "Technques and Knowledge used for adaption during *case*-based problem solving", in LNAN 1416, vol.2, Springer, 1998, pp: 497-506.

[27] Xia Q., Rao M. "Incorporating system dynamics in *case*-based reasoning for process operation support", IEEE int. conference on Systems Man Cybernetics and Simulation, Vol. 3, 1997, pp: 2075-2080.

[28] Yamamoto S, Kimura R, Miyahara H., "Automatic Knowledge Acquisition System for blast Furnace Burden Distribution Operation", IEEE Symposium on ETFA (Emerging Technologies and Factoy Automation), 1994, pp: 107-114.

[29] Colomer J, Melendez, J, De la Rosa J.L., Aguilar, J. "A qualitative/quantitative representation of signals for supervision of continuous systems", in ECC97, Brussels, 1997.

[30] Rengasamy, "A framework for integrating process monitoring, diagnosis and superviosy control", PhD. Thesis of the Purdue University, August, 1995.

[31] Meléndez 1998, "Integration of Knowledge-Based, Qualitative and Numeric Tools for Real Time Dynamic Systems Supervision", PhD. Thesis of the Universitat de Girona, 1998.