*Journal of Intelligent and Robotic Systems* **40:** 207–232, 2004.



# The Discrete Event Concept as a Paradigm for the "Perception-Based Diagnosis" of Sachem

MARC LE GOC
*LSIS, Domaine Universitaire St Jérôme, Avenue Escadrille Normandie–Niemen, 13397 Marseille Cedex 20, France, and*
*Sachem Consulting, Sollac Méditerranée, Bât DB25, 13776 Fos sur Mer Cedex, France;*
*e-mail: marc.legoc@lsis.org, marc.legoc@tixis.arcelor.com*

CLAUDIA FRYDMAN
*LSIS, Domaine Universitaire St Jérôme, Avenue Escadrille Normandie–Niemen, 13397 Marseille Cedex 20, France; e-mail: Claudia.frydman@lsis.org*

(Received: 8 August 2003; in final form: 30 December 2003)

**Abstract.** Sachem is an extensive large-scale real time knowledge-based system designed to monitor and diagnose blast furnaces. This paper aims at illustrating the way the concept of discrete event allowed the definition of a "perception-based diagnosis" approach as a recursive and holographic abstraction process of a discrete event. The example of the diagnosis of a "thermal load" phenomenon on a blast furnace is used in order to illustrate the way Sachem apply the "perception-based diagnosis" approach. Some considerations about the blast furnace and the development of Sachem are also presented in the paper to recall the complexity and the issue of the design of powerful perception systems.

**Key words:** fault diagnosis, monitored control systems, knowledge-based systems, discrete event systems, artificial intelligence.

## 1. Introduction

Sachem, an extensive large-scale real-time knowledge-based system designed to monitor and to diagnose dynamic processes, is an excellent example of Artificial Intelligence used to the means of industrial and economic success.

Sachem was initially developed to monitor and to diagnose the blast furnaces of Usinor, a company in the Arcelor group, the world's largest producer of steel products. The aim of Usinor was to save up to 1€ per ton of pig iron (see (Le Goc and Thirion, 1999) for a presentation of the project). Currently, Sachem earns between 1.5 and 1.7€ per ton of pig iron. With six blast furnaces equipped with a Sachem system, Arcelor earns between 16,5 and 18,5 millions euros per year.

Sachem works in real time and continuously (i.e., 24 hours per day, 365 days per year), with an availability ratio equals to 99,7%. The main task of Sachem is to reason in time and in space in order to determine the state of the blast furnace from the sensor data provided by the instrumentation (Figure 1).

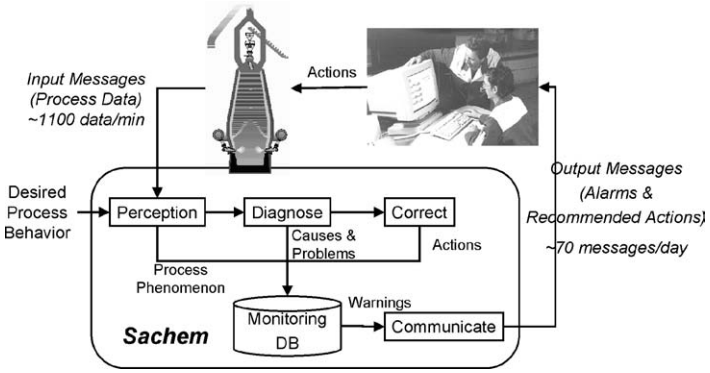


*Figure 1.* Sachem system in operation.

One of the main difficulties when designing such a system is the acquisition and the representation of the perception knowledge of experts that monitor and diagnose the blast furnace. The main properties of the perception knowledge are (i) to be "under-symbolic" and (ii) concerned with temporal and spatial evolutions of the process. The first property means that an expert cannot use words in order to describe its own cognitive process of perception. The second property means that humans have difficulties when describing precisely the evolutions within a system: it is easier to speak about the static properties than the dynamic properties. It is therefore very difficult to model the reasoning of a control process expert. The artificial intelligence approach stumbles over this difficulty.

This paper aims to illustrate the way the notion of discrete event allowed the design of the perception function of Sachem and its global reasoning process.

The next section introduces briefly the problematic of the blast furnace control. This presentation allows in Section 3 to analyze the structure of the knowledge required within a process control task. This analysis shows that the discrete event concept is a basic element for representing the process control knowledge for blast furnaces. In Section 4, the principles of the perception function of Sachem are described.

Section 5 aims at describing the main characteristics of Sachem. This section provides also some quantities that help in evaluating the scope of Sachem and the development project. The real example of the detection of a "scaffolding" phenomenon is used in Section 5 in order to describe the perception function of Sachem.

The paper concludes then on the economic performances of Sachem and introduces our currents developments for the next generation of Sachem systems.

## 2. Blast Furnace

Each of the two Fos sur Mer blast furnaces (Figure 2) represents an investment of 760 million euros for the first construction. A blast furnace is re-constructed every $\sim$15 years ($\sim$150 million euros).

*Figure 2.* Blast furnaces at Fos sur Mer.

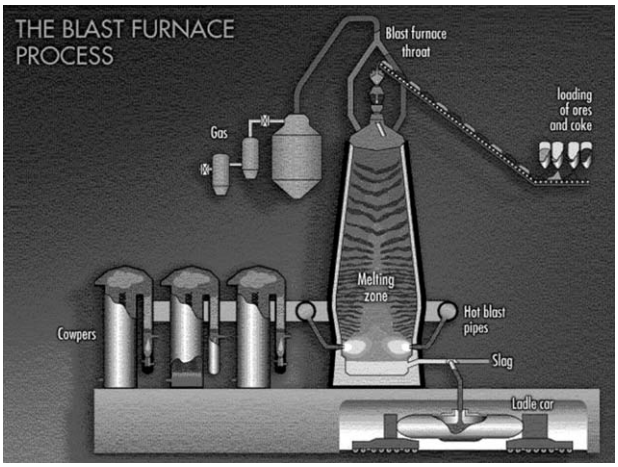


*Figure 3.* Blast furnace process.

## 2.1. BLAST FURNACE PROCESS

A blast furnace is a structure of ∼100 meters height that contains a chemical and thermal gas–solid counter-current reactor of ∼3,000 $m^3$ in ∼27 meters height (Figure 3). The internal pressure is ∼3 absolute bars and the power inside the reactor is ∼1,300 megawatts. Such a blast furnace produces around 6,000 tons of hot metal per day.

A blast furnace is loaded at the top (throat) with ore and coke (i.e., more than 95% of carbon). Hot blast pipes are used to inject hot air with additions of oxygen and pulverized coal. The oxygen burns the carbon contained in the coke and a flame of ∼ 2200°C is created. This combustion produces a gas (CO) that reduces the ores ($Fe_xO_y$). The heat and the reduction gas generate a flow of liquid iron and slag that casts down to the blast furnace crucible. There, the liquid iron charges with carbon to form the "pig iron", which is extracted through a hole. Slag and pig iron is then

separated by decantation, and pig iron is then carried to the steelworks with ladle cars.

### 2.2. SPECIFICITIES OF BLAST FURNACE

One of the main specificities of blast furnace is the lack of mathematical model of its internal dynamic.

There are at least two main reasons:

- The first is the complexity of the physical and chemical interaction phenomena between gas, solids and liquids. These interactions govern the behavior of a blast furnace and up to now, no mathematical model exists for describing these interactions in a whole.
- The second comes from to the extreme conditions that rules inside the blast furnace: very high temperatures, acidity, dust, high pressure, etc. With such operational conditions, it is industrially impossible to maintain sensors inside the blast furnace load. As a result, the thousands of instrumentation sensors are positioned around the blast furnace stack and no internal variable of the load is accessible to be measured.

The lack of mathematical models for the blast furnace dynamic led to the impossibility of applying the theories of the control process.

Another important specificity of blast furnace is the very long learning period (5–10 years) that is necessary to learn to control the production of blast furnace (10 persons are required per workstation). This is related to the lack of mathematical models for the blast furnace dynamic. The control of the production depends significantly on experience, that is to say practical skills and informal knowledge. An ageing workforce and the subsequent loss of experience due to social and economic conditions also affect this factor.

The elements described above constitute the main incentives to design the Sachem system through the means of artificial intelligence techniques.

## 3. Knowledge to Control

Every controller, human or system, performs two basic functions (Figure 4): the perception of the process state and the correction of the unsatisfactory states. The state of the process is then the pivot of the two basic functions of diagnosis and monitoring system: the perception function aims at recognizing the process state and the correction function defines the adequate actions to achieve. Diagnosis is then included in the perception function in order to complete the description of the process state.

The goal of the controller is precisely the disappearance of the unsatisfactory state(s): an action is required when the state of the process is not satisfactory, and otherwise nothing has to be done. So, when required, the controller must decide and achieve and action on the process.

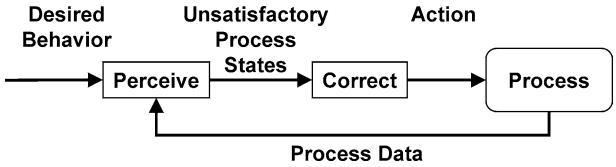


*Figure 4.* The perception–action closed loop.

An action is a modification of at least one of the input variables of the process. The causal relations between the variables will transform and propagate a modification of an input variable on the internal variables and ultimately on the output variables.

The blast furnace experts know a set of causal relations that are operational at least during the exploitation phases (i.e., specific causal relations are required for starting up or switching off the process). This type of causal knowledge presents the following characteristics:

- A modification that occurs inside the blast furnace is called a phenomenon.
- A phenomenon has effects on signals. The apparition of a phenomenon inside a blast furnace causes modifications on signals.
- A particular phenomenon causes a specific set of modifications on a specific set of signals. This specific set of modifications is called a "signature" (hallmarks).
- The detection of the occurrence of a phenomenon is achieved by the recognition of its effects on signals via an appropriate set of signatures.

Generally, the expert's knowledge refers to the "usual" effects of a modification (direction, magnitude and time interval) of the value of a variable on another. Typically, the magnitude of the effect is evaluated by the means of steady state models. Some knowledge about the context can clarify the conditions for using a known causal relation. These conditions are given in terms of the absence of other phenomenon that "inhibits" or prevents some causal relations.

### 3.1. CAUSAL KNOWLEDGE FOR CONTROL

To control the behavior of a process, one must then know the relations linking causes to effects. The natural form of the expert knowledge is the "if–then" rule. Conceptually, the simpler form of such a relation is:

> If the process is in the state $X$ and if the action $U$ is achieved
>
> Then the process produces the output $Y$.

In this rule, the process output vector $Y$ is the effect of a modification of the input vector $U$. This relation depends of the process state vector $X$: the condition for having the $Y$ effect from the $U$ modification is the fact that the process is in the

state $X$ ($X$, $U$ and $Y$ are vectors of the real number set $\Re$). The causal relation can be modeled as a 3-tuple of the form:

$$C(Y, X, U).$$

In order to clarify the key role of the state vector, this relation can be rewritten under the form of two binary causal relations:

$$C(X, U) \wedge C(Y, X),$$

where $C(X, U)$ denotes a causal relation from $U$ to $X$. The control of a process behavior consists in reversing the causal relations in order to reach a goal:

If the goal $G$ is to obtain the output $Y$ and if the process is in the state $X$
Then the action $U$ must be achieved.

The inverse relation introduces a new term: the goal $G$ of the controller. This inverse relation is then a 4-tuple of the form $I(Y, U, X, G)$ so that

$$I(Y, U, X, G) \Leftrightarrow C(X, U) \wedge C(Y, X) \wedge \text{Equal}(Y, G).$$

These causal relations are propositions about steady states of the process represented by vector values. They do not integrate the dynamic of the process, that is to say the way the process is changing. In order to evoke the change, the propositions must connect modifications of vectors (Villa, 1994).

The basic hypothesis used to design Sachem is that expert's reason mainly directly on state transitions rather than on the state of the process. Because state transitions can be represented by discrete events (Zeigler, 1976, 1984; Giambiasi, 1999), the knowledge to control the blast furnace can be represented with propositions about discrete events and temporal windows that constraints the time of the events.

### 3.2. CAUSAL KNOWLEDGE ABOUT DISCRETE EVENTS

The discrete event form of a causal relation adds timed constraints between the vector modifications:

If the process is in the state $X$ at time $t$
and if the action $\Delta U$ is achieved at the same time
Then the state process will transit to a new state $X'$
at time $(t + \Delta t_x)$ such that $\Delta t_x \in [T_x^-,\ T_x^+]$.

If the process transit from state $X$ to state $X'$ at $t$
Then the process will produce the output $\Delta Y$
at time $(t + \Delta t_y)$ such that $\Delta t_y \in [T_y^-,\ T_y^+]$.

The state transition from $X$ to $X'$ constitutes an event and is denoted $\Delta X$. The discrete event form of the causal relations is then (Travé-Massuyès and Milne, 1997):

$$C(\Delta X,\ \Delta U,\ [T_x^-,\ T_x^+]),$$
$$C(\Delta Y,\ \Delta X,\ [T_y^-,\ T_y^+]).$$

More generally, the form of the knowledge in order to control the blast furnace is the following:

$$R(\Delta\text{Out},\ \Delta\text{In},\ [T^-,\ T^+]),$$

where

- $R$ represents a relation,
- $\Delta$Out is the output event,
- $\Delta$In is the input event,
- $[T^-,\ T^+]$ is the timed constraint that defines the time window for observing the occurrence of the output event "$\Delta$Out" after the occurrence of the input event "$\Delta$In": if the input event occurs at time $t$, the output event would be observable in the interval $[t + T^-,\ t + T^+]$.

This knowledge means that there exist a relation $R$ that connect an output event $\Delta$Out with an input event $\Delta$In so that when the input event occurs at time $t$, the output event *would be* observed at time $[t + T^-,\ t + T^+]$ (Hanks and McDermott, 1994). The hypothetical flavor of this knowledge aims at catching the complexity of the blast furnace's behavior: the relation can be observed, in general, but there are specific situations where the relation cannot be observable. In other terms, phenomena can interact so that the output event can be unobservable in specifics circumstances.

### 3.3. KNOWLEDGE REPRESENTATION

A relation $R(\Delta\text{Out},\ \Delta\text{In},\ [T^-,\ T^+])$ is usually broken down into:

- a logical relation $R_{\text{L}}(\Delta\text{Out},\ \Delta\text{In})$.
- a timed constraint $R_{\text{T}}(\text{d}(\Delta\text{Out}),\ \text{d}(\Delta\text{In}))$, where "d($\Delta$Out)" and "d($\Delta$In)" are the event times of the events "$\Delta$Out" and "$\Delta$In". Here, the timed constraint means simply:
$$\text{d}(\Delta\text{Out}) - \text{d}(\Delta\text{In}) \in [T^-,\ T^+]).$$

Such a relation can be represented with a graph (Figure 5). It is to note that reasoning with the two types of relation, causal and temporal, is currently the one of the major difficulties when designing a diagnosis system (Cauvin et al., 1998).

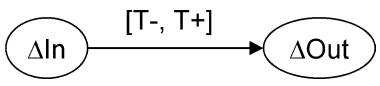


*Figure 5.* Relation between events.

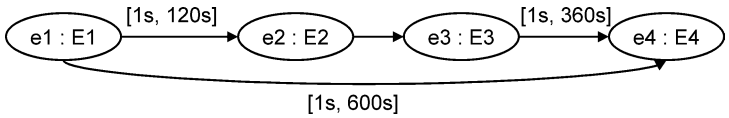


*Figure 6.* Example of signature.

Because the events can be classified in categories, the basic form of the Sachem knowledge is illustrated in Figure 6 and is called a "signature". This figure expresses that a sequence of 4 events $\{e_1, e_2, e_3, e_4\}$, respectively of type $E_1$, $E_2$, $E_3$ and $E_4$ are the sign of a specific modification iff:

$$\begin{aligned} &d(e_2) \in [d(e_1) + 1\,s,\ d(e_1) + 120\,s] \wedge d(e_3) > d(e_2) \\ &\quad \wedge d(e_4) \in [d(e_3) + 1\,s,\ d(e_3) + 360\,s] \\ &\quad \wedge d(e_4) \in [d(e_1) + 1\,s,\ d(e_1) + 600\,s]. \end{aligned}$$

The types $E_1$, $E_2$, $E_3$ and $E_4$ define particular values of attributes of event frames. The frames define categories of events under the form of logical properties. In other words, the basic knowledge to control a blast furnace is represented by a graph where nodes are restriction over event frames and links are timed constraints. A "signature" is then a graph of discrete events linked with temporal constraints.

Signatures are used in order to represent the conditions of perception of blast furnace phenomenon: a signature describes the start of a phenomenon and another one describes the end of a phenomenon. In order to detect a phenomenon, the signatures must be "operationnalized" (i.e., "compiled") in executable structures like "chronicles" (Ghallab, 1996).

Supplementary knowledge is required to determine the properties of the phenomenon like the category, the localization or the magnitude. Such knowledge is static and is represented by usual rules of the first order predicate calculus.

### 3.4. KNOWLEDGE OPERATIONALIZATION

Chronicle calculus is a framework for representing and using temporal constraints networks where nodes are events and arcs are temporal constraints. A temporal constraint manager, including a pattern matching mechanism, verifies if the occurrence of events satisfies the temporal constraints.

Chronicle calculus rely on propositional reified logic formalism (Soham, 1987) where the environment is described through domain attributes $P : v$ where $P$ is the attribute and $v$ its value (Dousson, 1996). Two predicates are used for temporally qualify propositions (Figure 7):

- "hold$(P : v, (t_1, t_2)$" represents persistency of the value of a domain attribute $P$ over an interval $[t_1, t_2]$ without knowing when this value was reached,
- "event$(P : (v_1, v_2), t)$" is a time stamped instance of event pattern corresponding to an instantaneous change of a domain attribute value $P$.

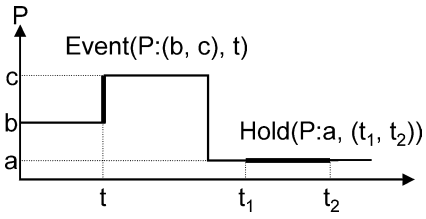


*Figure 7.* "Event" and "hold" predicates.

```
Chronicle SwitchProblem {
    //  - Forthcoming chronicle events
    event (Transmission: (on, off), e1) ;
    event (Transmission: (off, on), e2) ;
    event (Component1 : (?, ok), e3) ;
    event (Component2 : (?, ok), e4) ;
    event (Component3 : (?, ok), e5) ;
    //  - Assertions (context)
    hold(Traffic:normal, (e1, e6))
    //  - Temporal constraints
    e1  < e2 < e3 < e6;
    e2 < e4 < e6;
    e2 < e5 < e6;
    (e2 – e1) in [0, 180] ;
    (e6 – e2) in [60, 120];
    When recognized{
        report « Switch pb detection »
    }
}
```

*Figure 8.* Example of chronicle model.

A chronicle model represents a piece of the evolution of the world and is composed of four parts (Figure 8):

- a set of events which represents the relevant changes of the world for this chronicle,
- a set of assertions which is context of the occurrences of chronicle events,
- a set of temporal constraints which relates events and assertions between them,
- and a set of actions which will be processed when the chronicle is recognized.

The temporal constraints graph associated with a chronicle (Figure 9) allows computing the least constrained path between each couple of events that will be used during chronicle recognition process.

The recognition method is based on a complete forecast of forthcoming events predicted by chronicle model (Dousson et al., 1993). A temporal window $W$(e) is defined for each event e of the chronicle containing the possible occurrence dates for e in a partial instance of the chronicle. The actual date of e must be consistent

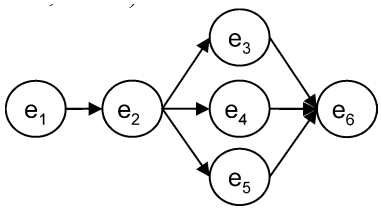


*Figure 9.* Temporal constraints graph of chronicle (without temporal constraints).

with constraints and known dates of the other events of the partial instance. The chronicle recognition evolves when a new event arrives or when time passes so that temporal constraints become obsolete.

The chronicle calculus technique is embedded in KOOL-94, the knowledge representation language used to implement Sachem (see further). This language integrates a temporal constraints manager derived from the IxTeT system (Dousson, 1996), which as also been integrated in the TIGER system (Milne et al., 1994; Aguilar et al., 1994; Travé-Massuyès and Milne, 1997). Other industrial examples of such techniques can be found in (Laborie and Krivine, 1997; Bredillet et al., 1994) for AUSTRAL or (Cauvin et al., 1998) for GASPAR.

The interested reader can refer to (Villa, 1994) for a survey of temporal reasoning in artificial Intelligence and to (Dague, 2001) for an introduction to the model based diagnosis theory and the references for the main diagnostic systems.

## 4. Spatial Discretization

The aim of the experts' knowledge is to transform the measurements (i.e., real numbers) into symbols (i.e., natural numbers) so that specific reasoning can be achieved. This transformation is called "quantization" (Zeigler et al., 2000) and is concerned with the "number to symbol" or the "quantity to concept" transformation.

For Sachem, a recursive discrete event generation process that constitutes an abstraction process realizes this transformation. The recursive process starts at the signal level: the temporal evolutions of the signals are described by a flow of discrete event. This flow of discrete event is then interpreted according to particular chronicles in order to recognize new discrete events of more abstract level.

This section illustrates this discrete events abstraction.

### 4.1. DISCRETE EVENT FLOW

The discrete event generation mechanism is based on a state space discretization (Figure 10). A set of static or dynamic thresholds is used in order to define space areas. Basically, an event is generated when a signal crosses a threshold that is to say when the process move from one space area to another space area.

In the example of Figure 10, 4 thresholds ($-1, -2, +1$ and $+2$) are defined for a variable $x_i(t)$ (idem for its derivative $\dot{x}_i(t)$). This defines 5 ranges called

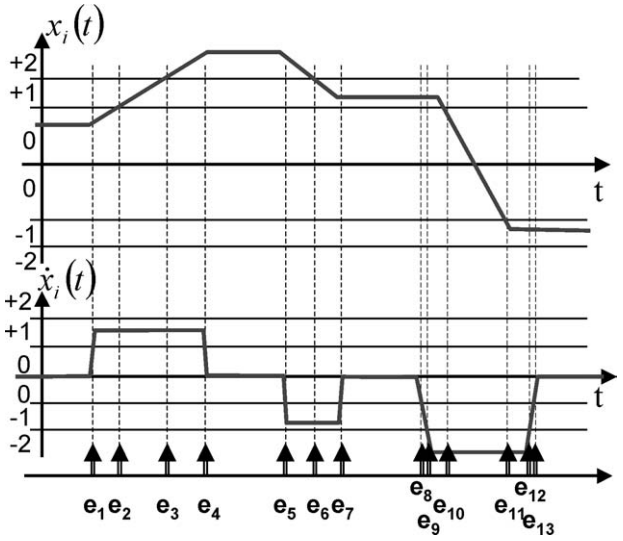


*Figure 10.* Discrete event generation.

$$
\begin{aligned}
e_1 &\equiv (t_1, \dot{x}, 1) &&\leftrightarrow (x = ?, \dot{x} = 1),\\
e_2 &\equiv (t_2, x, 1) &&\leftrightarrow (x = 1, \dot{x} = 1),\\
e_3 &\equiv (t_3, x, 2) &&\leftrightarrow (x = 2, \dot{x} = 1),\\
e_4 &\equiv (t_4, \dot{x}, 0) &&\leftrightarrow (x = 2, \dot{x} = 0),\\
e_5 &\equiv (t_5, \dot{x}, -1) &&\leftrightarrow (x = 2, \dot{x} = -1),\\
e_6 &\equiv (t_6, x, 1) &&\leftrightarrow (x = 1, \dot{x} = -1),\\
e_7 &\equiv (t_7, \dot{x}, 0) &&\leftrightarrow (x = 1, \dot{x} = 0),\\
e_8 &\equiv (t_8, \dot{x}, -1) &&\leftrightarrow (x = 1, \dot{x} = -1),\\
e_9 &\equiv (t_9, \dot{x}, -2) &&\leftrightarrow (x = 1, \dot{x} = -2),\\
e_{10} &\equiv (t_{10}, x, 0) &&\leftrightarrow (x = 0, \dot{x} = -2),\\
e_{11} &\equiv (t_{11}, x, -1) &&\leftrightarrow (x = -1, \dot{x} = -2),\\
e_{12} &\equiv (t_{12}, \dot{x}, -1) &&\leftrightarrow (x = -1, \dot{x} = -1),\\
e_{13} &\equiv (t_{13}, \dot{x}, 0) &&\leftrightarrow (x = -1, \dot{x} = 0).
\end{aligned}
$$

*Figure 11.* Discrete event flow.

$-2, -1, 0, +1$ and $+2$. Whenever $x_i(t_k)$ (or $\dot{x}_i(t_k)$) crosses a threshold, an event is generated. For example, at time $t_1$, $\dot{x}_i(t_1)$ crosses the "+1" threshold. An event $e_1 \equiv (t_1, \dot{x}, 1)$ is generated signifying that $\dot{x}_i$ crosses the threshold "+1" at time $t_1$.

This discrete event generation mechanism produces a discrete event flow that describes the evolution of the process state trajectory over time (Figure 11). The discrete event flow is a suite of discrete event describing the times of the entry in a new space area.

The discrete event flow defines a piece wise constant trajectory (Figure 12) that constitutes a model of the process state trajectory. The two representations

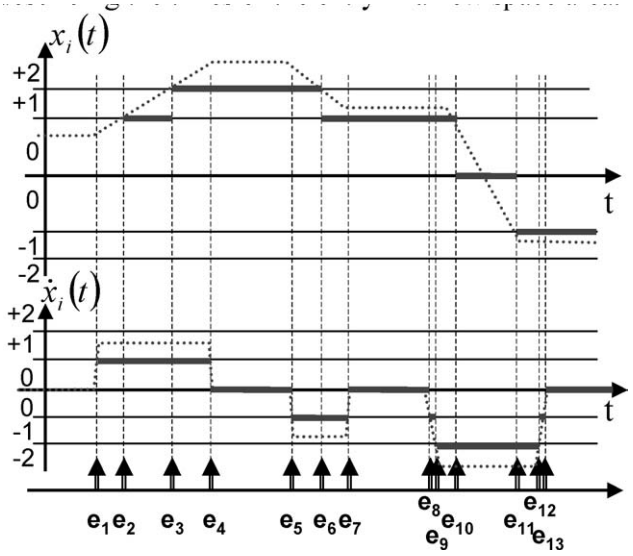


*Figure 12.* Piecewise constant trajectory.

are equivalent: the piecewise constant trajectory is a continuous model while the discrete event flow is a discrete event model and both are continuous time models.

This mechanism leads then at a temporal segmentation. This is similar to a temporal sampling with a sampling period that is not constant.

### 4.2. TRAJECTORY INTERPRETATION

In order to interpret the discrete event flow, the discrete events can be positioned in a phase plane. Figure 13 illustrates the interpretation of the event flow of Figure 11 in a phase space of Poincaré.

The interpretation principle of the discrete event flow consists in considering that a discrete event defines a point in the phase plane. The continuous process state trajectory can then be reconstructed when interpreting the points defined by the events as the tangent point of the process state trajectory: at the time of the discrete event, the process state trajectory coincides with the point defined by the event. Then linking the tangent points with a curve produces an interpretation of the process state trajectory (Figure 13). This approach has similarities with the Generalized-Devs approach (Giambiasi et al., 2000).

The experts define a particular trajectory corresponding to an evolution with a meaning of particular interest. The interpretation principle allows the representation of the expert's knowledge under the form of a succession of discrete event type that constitutes a signature. Such a signature is represented with a chronicle for the signature recognition.

For example, the sequence $e_8$ to $e_{13}$ shows a significant decrease of $x_i(t)$ (Figure 13). The problematic aspect of such a trajectory is the succession of an event

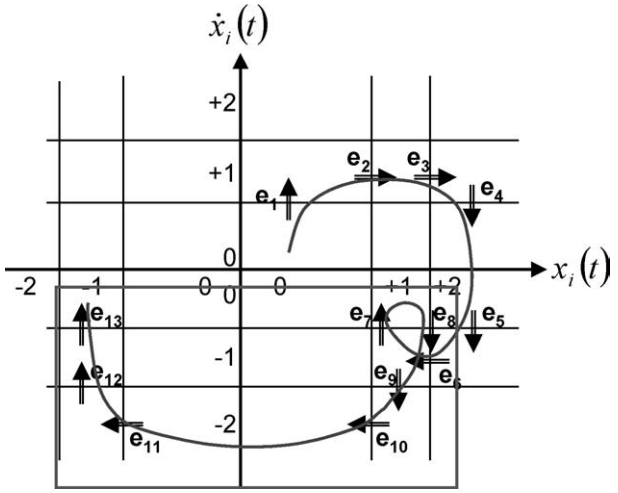


*Figure 13.* Trajectory interpretation.

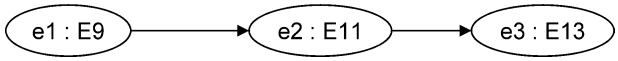


*Figure 14.* Example of problematic sequence.

of the same type as $e_9$, labeled $E_9$, followed by an event of the type of $e_{11}$ ($E_{11}$), and the latter followed by an event of $E_{13}$ type (Figure 14).

This situation, with the appropriate temporal constrains, means that a very low-level ($E_{13}$) has been reached with an almost null derivative ($E_{11}$) after a strong decrease ($E_9$). If the goal is to have a normal level, something must be done.

The recognition of this significant segment in a discrete event flow uses the signatures defined by the experts. Whenever a sequence of events satisfies the logical and timed constraints of the chronicle that represents a signature, a new discrete event is generated.

### 4.3. RECURSIVITY PROPERTY

The quantization process generates the first level of discrete event. From this first level, the signature recognition process generates a second level of discrete event. The events of this second level are more abstract than those of the first level.

One the advantages of the discrete event paradigm is that this representation allows the design of a recursive process: the discrete event of the second level can also be used in order to recognize signature of another type. This second stage of recognition will generate a third level of discrete events that can again be used for a third stage of signature recognition, and so on. The "perception-based diagnosis" approach leads then to a recursive and holographic design. The process behavior is described at different abstraction level with a unique paradigm, the discrete event paradigm, and the transition from a level of abstraction to another is based on a sim-

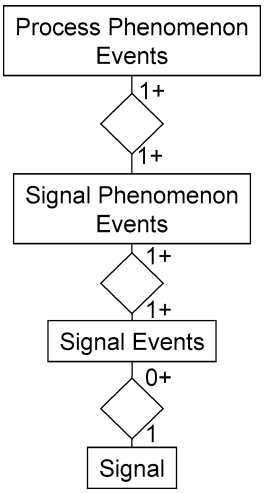


*Figure 15.* The three levels of discrete events.

ilar signature recognition process. As a consequence, at each level of abstraction, the reasoning process is similar.

For example, the knowledge of Sachem defines 3 levels of discrete events (Figure 15):

- The "Signal Event" level. At this level, an event is connected with the matching of a behavior model and a signal.
- The "Signal Phenomenon" level. This level of event describes a particular evolution of the signal, according to a "norm" (i.e., a signature). This is a couple of Signal Event, detected on the same variable, one that times the start of the Signal Phenomenon and one for the end.
- The "Process Phenomenon" level. This level aggregates different signal events generated from different variables in order to recognize the specifics effects of a phenomenon. A Process Phenomenon is also a couple of events, one for the beginning of the phenomenon and one for the end, the time comes from the signal events uses to recognize the process phenomenon.

The recursion property is interesting for many different aspects, in particular, for the representation of temporal knowledge and the design of monitoring and diagnosis cognitive agents. The same paradigm can be used to represent the knowledge required for different stages of recognition process.

The next section illustrates this approach with the example of the detection of a "scaffolding" phenomenon by Sachem.

## 5. Perception of a Phenomenon

The first level of the discrete event generation of Sachem is applied to signals and is achieved with Signal Processing algorithms and artificial neural networks (Le Goc et al., 1998).

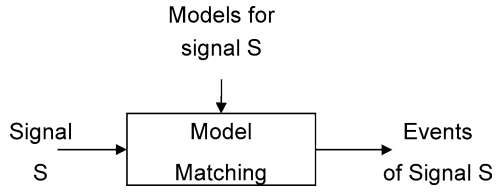


*Figure 16.* Model matching.

### 5.1. NOTION OF SIGNAL EVENT

The principle is based on the matching of a segment of a signal with a model (Figure 16).

Three types of events are required:

- "Start event". This event indicates that the behavior of a signal $S$ matches a particular model. Such an event is generated when the quality coefficient of the matching is greater than the required threshold.
- "End event". This event indicates that the current model no longer matches. This event is the opposite of the "matching event".
- "Lost event". This event indicates that the matching algorithm is unable to decide if a model matches or not. This case occurs when data are not present or are invalid.

These three types of events are called "signal events" and constitute the basis for the perception analysis. The information concerned with the generation of the event (sensor, space area, signal processing algorithm, nature of the event, etc.) is associated with the event in a "signal event object" (cf. the technical design section).

### 5.2. SIGNAL EVENT PERCEPTION

In Figure 17, the events are produced with a real time linear regression algorithm applied to the "Thermal Load" (TL) variable. This variable measures the energy that flows out the blast furnace via stack.

The $r^2$ variable measures the quality of the linear regression. The matching is achieved when this quality coefficient is over a minimum threshold, typically 0.8, and when the coefficients ($a_i$ and $b_i$) of the model respect the values defined by the experts.

The first step of the discrete event generation is the production of the "Start" and "End" events when a signal matches with a model (Figure 17).

The next step determines the actual time of the evolution. Because the analysis is achieved in real time, the event detection introduces a delay in the signal analysis. When an event is detected, the signal is analyzed from the present to the past, in order to readjust the actual model parameters (Figure 18).

The signal events are then generated with 2 times, the discrete event detection time and the start time ($ts_i$) or the end time ($te_j$) of the signal behavior. Be-

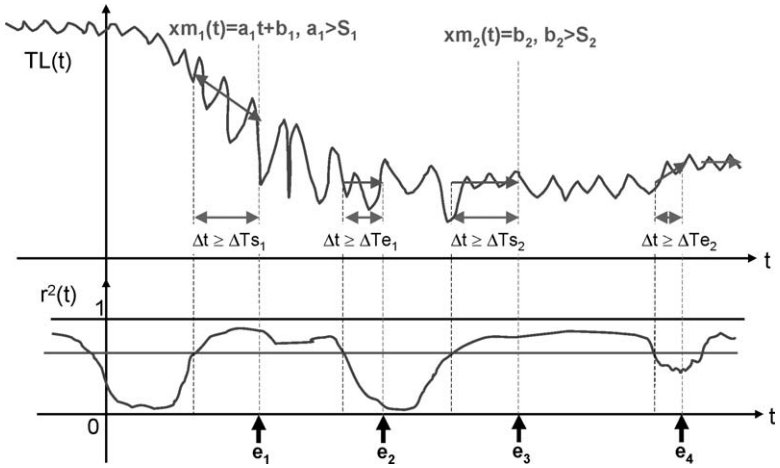


*Figure 17.* Step #1 model matching.

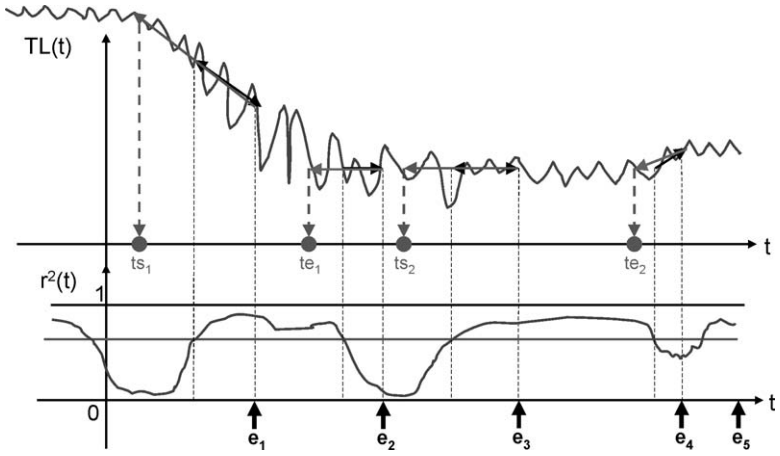


*Figure 18.* Step #2 signal event generation.

cause a discrete event is coded as an object within Sachem, every discrete event instances contains all information concerning the pattern matching (signal identity, the matched model with the current parameters, etc.).

### 5.3. SIGNAL PHENOMENON PERCEPTION

The flow of signal events is then analyzed according to the set of signatures for the TL signal. For example, the reception of the sequence $\{e_1, e_2, e_3, e_4\}$ may satisfy a "thermal load stability" signature (Figure 19). Sachem will verify the timed constraints ($\Delta T_i \geqslant ST_i$) and the values of the parameters ($\alpha_1$ and $\beta_2$).

If these constraints satisfy the requirements, a "thermal load stability" signal phenomenon will be generated. The beginning of this signal phenomenon will be

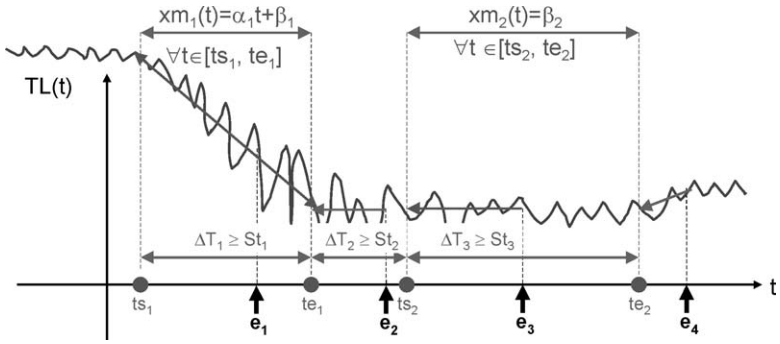


*Figure 19.* Step #3 event flow analysis.

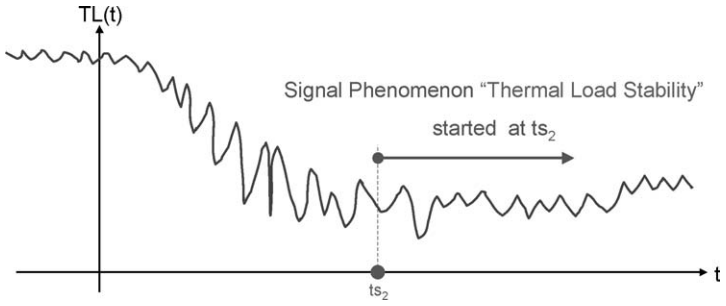


*Figure 20.* Step #4 trajectory interpretation.

timed at $ts_2$ (Figure 20). The event data are used to analyze the evolution of a signal phenomenon (qualification of the magnitude, localization according to the geometrical position of the sensors, etc.). In the same way, Sachem determines the end of a signal phenomenon with dedicated chronicles.

### 5.4. PROCESS PHENOMENON PERCEPTION

The flow of signal phenomenon is then analyzed in order to identify the Process Phenomenon that occurs inside the blast furnace (Figure 21).

In the example, the signature of the "Scaffolding" process phenomenon will be used to detect a new instance of this phenomenon that starts at time $ts_2$. Such a signature will merge 3 signal phenomena in order to decide the generation of the instance of the "Scaffolding" process phenomenon.

A process phenomenon has a specific signal phenomenon for the detection of its beginning and another one for the detection of its end. Between these two times, the signal phenomenon flow is analyzed in order to qualify the evolution of the phenomenon.

For some process phenomenon, the 3 main operations (the detection of the beginning, the qualification of the evolution and the detection of the end) must merge the signal phenomenon in the spatial dimension.

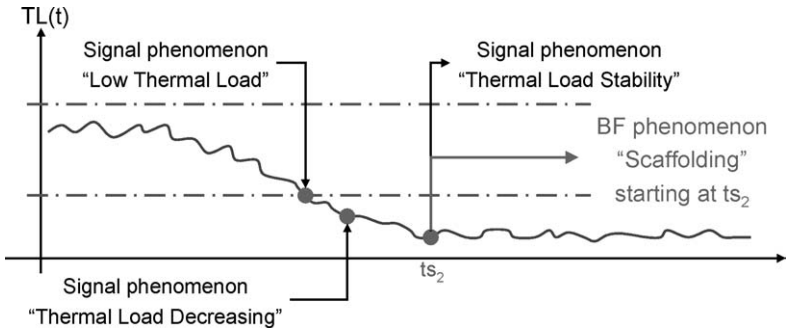


*Figure 21.* Step #5 process phenomenon detection.

For example, the wall temperature of a blast furnace stack is measured through a matrix of $8 \times 13$ virtual sensors (a virtual wall temperature sensor is an aggregation of at least 3 physical sensors). The detection of the beginning of a "high level temperature on the wall" requires the aggregation of at least 4 neighboring sensors. The phenomenon is then located within the matrix and the spatial evolution is analyzed along time: a phenomenon can increase and merge with another instance, or disappear.

## 6. Sachem Overview

Sachem describes the current physical phenomena, that work within the blast furnace and qualifies them in terms of alarms (or warnings) according to the current operational process context. The aim of the alarms is to warn human-operators about the unsatisfactory behaviors of the blast furnace in order to avoid anomalies or incidents in the short and mid-term.

### 6.1. THE GLOBAL REASONING PROCESS OF SACHEM

The Sachem perception function describes thus the state of a blast furnace as a flow of phenomena that start, end at particular times and evolve over time and space as the process itself evolves.

Sachem organizes the operating phases of a blast furnace in "context" that allows to qualify each instance of phenomenon in terms of alarm, warning or "not problematic" (this is a value of the "severity" slot of a phenomenon instance). When a phenomenon is an alarm or a warning, a message is sent to the human-operators (Figure 22). A "context" considers the absence of particular instances of phenomenon over temporal periods. Such periods characterize the more or less good quality of the process control (Albers and Ghallab, 1997).

The graphical user interface of Sachem is divided into 3 parts: the instrumentation status (left), the messages that describe the current phenomena (upper right) and the messages recommending corrective actions (lower right). By clicking on the messages, the user accedes to the views that justify Sachem's decisions through

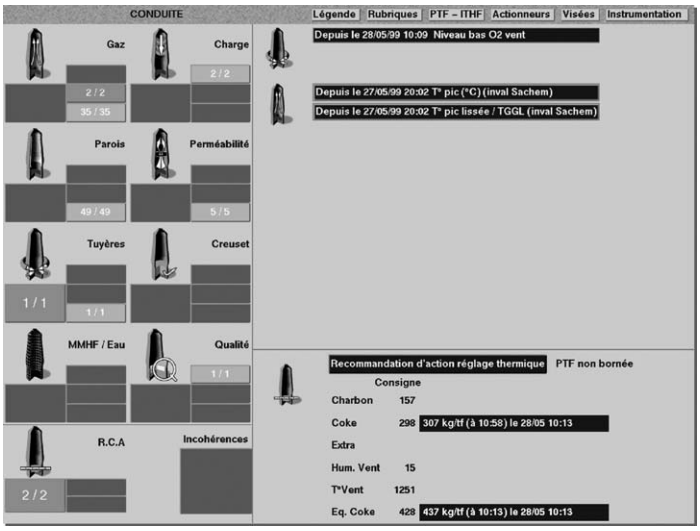


*Figure 22.* Sachem user interface.

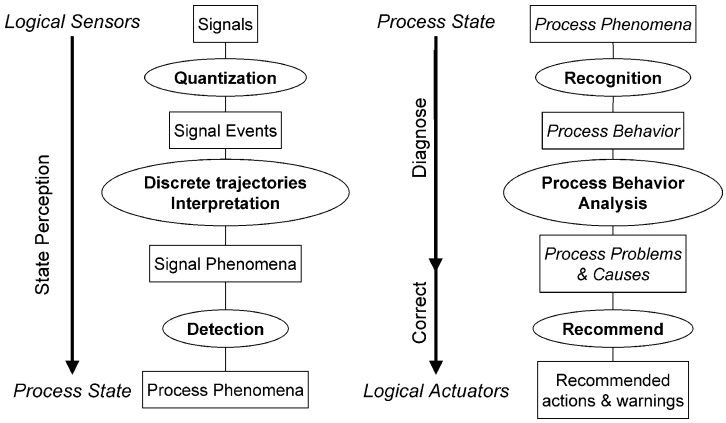


*Figure 23.* The global reasoning process of Sachem.

a set of curves representing the evolution of the main variables that are concerned with the meaning of the message.

The messages are displayed until the underlying problem disappeared (i.e., the phenomenon ends or the action is no more required). The whole discrete event based abstraction process described in this paper aim at managing the messages that are concerned with the phenomenon working inside the blast furnace.

The global reasoning process of Sachem is described in Figure 23. The perception function aims at describing the state of the blast furnace over time. The perception function of Sachem supplies a "diagnosis" function to complete the description of the state of the controlled process (Figure 23). This function analyzes the flow of phenomena in order to recognize the problematic behaviors, to identify

the hypothetical causes that explain the recognized behavior, and to advise which unsatisfactory states the process could potentially reach in the future. The diagnosis function is currently being designed with the same recursive approach.

### 6.2. SACHEM ARCHITECTURE

Figure 24 shows the architectural principle of the Sachem system that supports the implementation of the global reasoning process of Sachem. This architecture is set up to process more than 1,100 data items per minute for a given blast furnace.

According to this model, Sachem executes the following main tasks:

- Data acquisition, synchronization, and verification of the 1,100 data per minute.
- Data ordering, processing, validating (using physical and chemical models) and diagnosing to identify the damaged sensors. This function processes up to 4,500 data each minute in order to generate the signals to be analyzed.
- Signal analysis and pattern recognition, including neural networks, to detect signal modifications along the dimensions of time and space. This function performs a temporal segmentation of around 500 signals to produce the first flow of discrete events ("Signal Event").
- Phenomena detection. The flow of discrete events created from the signals is analyzed each minute in order to recognize specific behaviors that indicate the presence of phenomena. Each phenomenon is localized in the blast furnace and its spatial and temporal magnitude is qualified according to a specific set of ranges (typically; very low, low, normal, high, very high). According to the set of current phenomena that defines the context, each phenomenon can be qualified as warnings or alarms in order to alert the human-operators.
- Action recommendation for the human-operators, when a situation must be corrected. The requirements for correction are evaluated for each phenomenon and an energy balance is computed. The situation in its entirety is then evaluated in order to determine the adjustment of the actuators (magnitude and time).

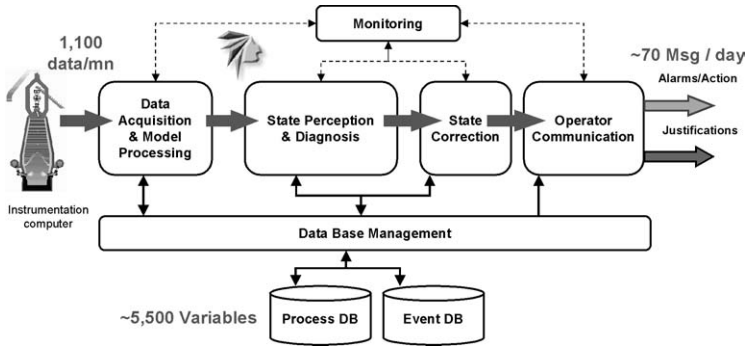


*Figure 24.* Sachem organic architecture.

- A user-friendly and context-dependent man machine interface transmits the warnings and the alarms to the human-operators and presents all the relevant information that is necessary to understand the importance of the phenomenon in the current situation.

The development of Sachem (1991–1998) represents an investment of 30 million euros (∼150 man × year). The team was composed of 20 to 30 engineers (including 6 knowledge engineers) and 12 experts. These considerations lead to methodological problems.

### 6.3. SACHEM METHODOLOGY

According to A. Newell definition, Sachem is a reactive rational agent, called a "Monitoring Cognitive Agent" that collaborates with human-operators in the control room (Figure 25) (Le Goc and Gaeta, 2003).

The CommonKADS methodology provides a general framework for the development of cognitive agent as knowledge-based systems (Schreiber et al., 2000; Breuker and Van de Velde, 1994).

This framework proposes guidelines to build an informal model, known as the "conceptual model" of the system, which must be completed to become a specification model. So the "conceptual specification" phase that produces the conceptual model complete the classical "functional specification" phase (producing the specification model) and the technical design (producing the design model) (Figure 26).

The knowledge-based system development can then be viewed as the addition of a knowledge acquisition process to the classical software development process (Le Goc et al., 2002).

The CommonKADS conceptual model is built in an implementation independent way. This model is based on the identification of different types of knowledge, which are distinguished as three different layers within the KADS conceptual model:

- The domain layer is the lowest layer. It contains knowledge about the application domain of the system, that is, the concepts and their relationships, used as roles in the next layer, and the reasoning rules. This knowledge is described

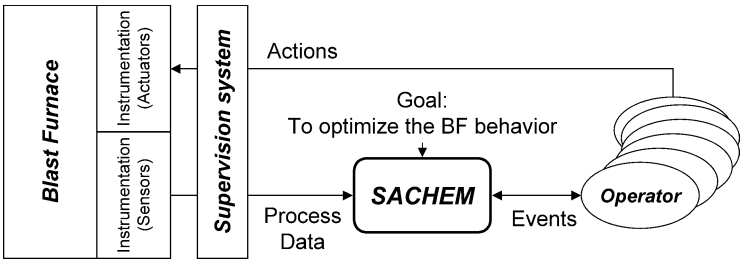


*Figure 25.* Sachem is a monitoring cognitive agent.

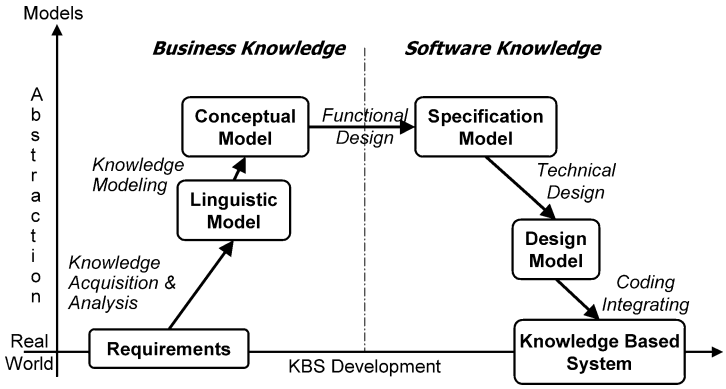


*Figure 26.* Model transformation of "SachemKADS" methodology.

freely from behavioral considerations; that is, from the way this knowledge will be used. Behavioral information is introduced in the other layers.

- The inference layer specifies the basic inference steps with inference structures. An inference structure is composed of inferences and roles and describes the transformations that the use of the domain knowledge can operate.
- The task layer is composed of a hierarchically organized set of tasks. Each task represents a problem-solving procedure. It is composed of inferences and/or subtasks and it specifies the control of their activation (sequence, iteration, conditional statement, etc.).

The conceptual model of the knowledge specifies the entire system. This model contains 25,000 objects for 33 goals, 27 tasks, 75 inference structures, 3200 concepts and 2000 relations. This represents 14 man-years of work for a team of 6 knowledge engineers and 12 experts during a period of 3 years.

The conceptual model is two parts: a generic model of interpretation, and a linguistic model, which is a representation in natural language of the domain knowledge. This linguistic model is made of 2000 pages of text and graphs, which are distributed in over 70 documents (Figure 26). The explicitation of the linguistic model is one of the modifications that have been done in order to adapt the CommonKads methodology for the Sachem development.

## 6.4. TECHNICAL DESIGN

The Sachem conceptual model has been implemented in 21 knowledge bases in order to recognize (i.e., to detect, to localize and to confirm) around 175 classes of phenomena.

All these items of knowledge are implemented with KOOL-94, a powerful language that combines concepts of:

- Objects (classes and meta-classes, methods and daemons).

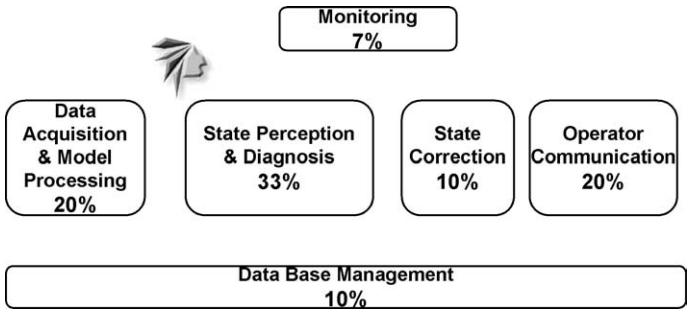


*Figure 27.* Knowledge distribution.

- First order logic rules. The deductive reasoning process is driven by the data with forward and backward chaining algorithms.
- Chronicle. A temporal manager based on the Allen logic analysis the flow of events to construct the sequences that satisfies the chronicles. The chronicles are used to implement the expert's signatures.
- Procedures. A C-like language permits the management of the instances that constitute the memory of the system and the communication with other software and the environment.

The Sachem knowledge bases contain more than 1060 classes of objects, 1100 first order logic rules and 140 chronicles.

The total software volume represents around 400,000 lines of code. About 60% of this amount of code implements the Sachem functional model, 33% of which is dedicated to the knowledge bases (Figure 27).

### 6.5. SACHEM ECONOMIC RESULTS

Figure 28 shows the effect of using Sachem on the frequency of incidents.

The frequencies have been measured during 1996 on the two blast furnaces of Fos-sur-Mer, which are identical, but only one was equipped with Sachem. The original level is the reference level obtained during 1995.

The incidents are classified in 3 types (low hot metal temperature, thermal loss and burden descent). Figure 28 shows that Sachem decreased the global number of incidents under of the expected level and totally avoided the most serious type of incidents: a low level of the hot metal temperature (which can lead to a destruction of the blast furnace). Sachem earns currently between 1.5 and 1.7€ per ton of pig iron. With six blast furnaces equipped with a Sachem system, Arcelor earns between 16,5 and 18,5 millions euros per year.

The technical and economical success of Sachem has led to the design of "Sachem-like" systems to assist the control of other industrial production processes. And today, Sachem is a commercial software product available on the world market.

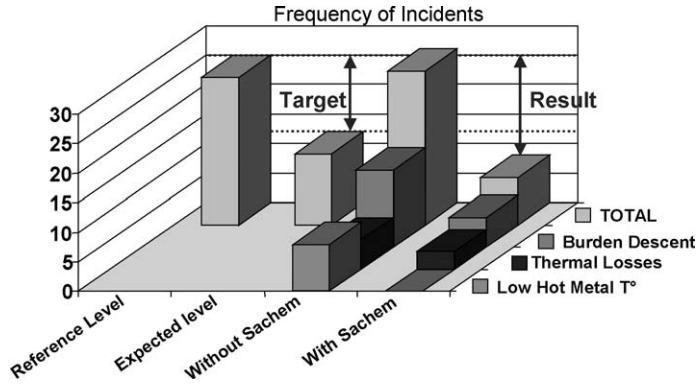


*Figure 28.* Sachem decrease the frequency of incident.

## 7. Conclusion

Sachem describes the current physical phenomena, and qualifies them in terms of alarms or warnings according to the current operational process context. The aim of alarms is to warn human-operators about the unsatisfactory behavior of the blast furnace in order to avoid anomalies or incidents in the short and mid-term.

The basis of the Sachem reasoning is the recognition of signatures that are represented with chronicles, a set of timed constraints linking a set of discrete event types. The discrete event paradigm allows the design of a recursive discrete event abstraction process based on the recognition of signatures.

We called this anthropomorphic approach of the process control the "perception-based diagnosis". Sachem analyzes continuously a flow of discrete events of more and more abstract level. The "perception-based diagnosis" is recursive and holographic: the process behavior is described at different abstraction level with a unique paradigm, the discrete event paradigm, and the transition from a level of abstraction to another is based on a similar signature recognition process. As a consequence, at each level of abstraction, the reasoning process is similar.

One of the advantages of the discrete event paradigm is the compactness. For example, one year of blast furnace represents around 30,000 process phenomena in a set of text file of 4.7 mb. This discrete event flow is the discrete event abstraction of a database of 235,000 mb containing the blast furnace raw data. The compactness factor is then 50,000.

Sachem does not implement the perception task of experts, which is a too complex and mysterious cognitive process to be described. However, with the same input data flow, the techniques implemented in Sachem produces the same output data as the best Arcelor expert in the Sachem skill area.

But the introduction of Sachem in the control room had an intriguing effect: the knowledge concerning the causal relations needed to be revised by experts. This revision is the result of the inviolable dating rules that are systematically and rigorously applied by Sachem in order to time the phenomenon (start and end).

Generally speaking, the temporal relations between phenomena must be updated, and in some cases, even the direction of the relation must be revised.

To this aim, we are now developing an approach to assist the expert in the discovering process of signatures at the process phenomena abstraction level. Our approach is based on the Markovian representation of a discrete event flow.

Integrated in a Java environment called the "ELP Lab", a set of tools has been developed to compute correlation between the beginnings of process phenomenon. The ELP language is a high-level knowledge representation language of signatures (Frydman et al., 2001). The ELP signatures are operationnalized with the DEVS formalism: the chronicles are translated into DEVS models that a DEVS simulator uses in order to recognize the discrete events sequences that satisfy the signatures. Our current work aims at defining a method to infer an ELP signature from a set of significant sequences.

## Acknowledgement

The authors would like to thank N. Giambiasi for helpful comments on earlier drafts and valuable discussions.

## References

Aguilar, J., Bousson, K., Dousson, C., Ghallab, M., Guash, A., Milne, R., Nicol, C., Quevedo, J., and Travé-Massuyès, L.: 1994, TIGER: Real-time assessment of dynamic systems, *Intelligent Systems Engrg.*, 103–124.

Albers, P. and Ghallab, M.: 1997, Context dependent effects in temporal planning, in: Steel and Alami (eds), *Recent Advances on AI Planning*, Lecture Notes in Computer Science 1348, Springer, New York, 1997, pp. 1–12.

Bredillet, P., Delouis, I., Eyrolles, P., Jehl, O., Krivine, J.-P., and Thiault, P.: 1994, The AUSTRAL expert system for power restoration on distribution systems, in: *Proc. of ISAP'94*, EC2 edn, pp. 295–302.

Breuker, J. and Van de Velde, W.: 1994, *The CommonKADS Library for Expertise Modeling*, IOS Press.

Cauvin, S., Cordier, M.-O., Dousson, C., Laborie, P., Lévy, F., Montmain, J., Porcheron, M., Servet, I., and Travé, L.: 1998, Monitoring and alarm interpretation in industrial environments, *AI Communications* **11**(3/4), 139–173.

Dague, P.: 2001, Théorie logique du diagnostic à base de modèles, in: *Diagnostic, Intelligence Artificielle, et Reconnaissance des Formes*, Hermes Science Publications, Paris, pp. 17–105.

Dousson, C.: 1996, Alarm driven supervision for telecommunication network: II. On-line chronicle recognition, *Ann. Télécommunications* **51**(9/10), 501–508.

Dousson, C., Gaborit, P., and Ghallab, M.: 1993, Situation recognition: Representation and algorithms, in: *Thirteenth Internat. Join Conf. on Artificial Intelligence*, Chambéry, France, pp. 166–172.

Frydman, C., Le Goc, M., Torres, L., and Giambiasi, N.: 2001, Knowledge-based diagnosis in Sachem using DEVS models, in: Tag Gon Kim (ed.), *Special Issues of Transactions of Society for Modeling and Simulation International (SCS), Recent Advances in DEVS Methodology* **18**(3), 147–158.

Ghallab, M.: 1996, On chronicles: Representation, on-line recognition and learning, in: Aiello, Doyle, and Shapiro (eds), *Proc. Principles of Knowledge Representation and Reasoning*, Morgan-Kauffman, 1996, pp. 597–606.

Giambiasi, N.: 1999, Abstractions à événements discrets de systèmes dynamiques, *RAIRO APII (Automatique–Productique Informatique Industrielle), Journal Européen des Systèmes Automatisés* (January 1999).

Giambiasi, N., Escude, B., and Ghosh, S.: 2000, Generalized discrete event specifications: G-DEVS coupled models, in: *Congres SCI 2000*, Orlando, USA, July 2000.

Hanks, S. and McDermott, D.: 1994, Modelling a dynamic and uncertain world I: Symbolic and probabilistic reasoning about change, *Artificial Intelligence* **66**, 1–55.

Laborie, P. and Krivine, J.-P.: 1997, Automatic generation of chronicles and its application to alarm processing in power distribution systems, in: *Internat. Workshop on Principles of Diagnosis (DX '97)*, Mont-Saint-Michel, France, pp. 61–68.

Le Goc, M., Frydman, C., and Torres, L.: 2002, Verification and validation of the Sachem conceptual model, IJHCS, *Internat. J. Human Computer Studies* **56**(2), 199–223.

Le Goc, M. and Gaéta, M.: 2003, Modelling structures in generic space, a condition for adaptiveness of monitoring cognitive agent, in: *AIS'2002, Artificial Intelligence, Simulation and Planning in High Autonomy Systems*, Lisbon, Portugal, 7–10 April 2002, *Special Issue of the J. Intelligent Robotics Systems*, to be published.

Le Goc, M. and Thirion, C.: 1999, Using both numerical and symbolic models to create economic value: The Sachem system example, in: *Proc. of the 27th McMaster Symposium on Iron and Steelmaking*, Hamilton, ON, Canada.

Le Goc, M., Touzet, C., and Thirion, C.: 1998, The Sachem experience on ANN application, invited paper at: *Neurap'98, 4th Internat. Conf. on Neural Networks and their Applications*, Marseille, France, pp. 315–321.

Milne, R., Nicol, C., Ghallab, M., Trave-Massuyes, L., Bousson, K., Dousson, C., Quevedo, J., Aguilar-Martin, J., and Guasch, A.: 1994, TIGER real time situation assessment of dynamic systems, *Intelligent Systems Engrg. J.* **3**(3), 103–124.

Newell, A.: 1982, The knowledge level, *Artificial Intelligence* **18**, 87–127.

Schreiber, G., Hakkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., and Wielinga, B.: 2000, *Knowledge Engineering and Management – The CommonKADS Methodology*, MIT Press, Cambridge, MA.

Soham, Y.: 1997, *Reasoning about Change: Time and Causation from the Standpoint of Artificial Intelligence*, MIT Press, Cambridge, MA.

Travé-Massuyès, L. and Milne, R.: 1997, TIGER: Gas turbine condition monitoring using qualitative model based diagnosis, *IEEE Expert Intelligent Systems Appl.* (May/June 1997), 22–31.

Vila, L.: 1994, A survey on temporal reasonning in artificial intelligence, *AICOM* **7**(1), 4–28.

Zeigler, B.: 1976, *Theory of Modeling and Simulation*, Wiley, New York.

Zeigler, B.: 1984, *DEVS Multifaceted Modeling and Discrete Event Simulation*, Academic Press, London.

Zeigler, B., Kim, T. G., and Praehofer, H.: 2000, *Theory of Modeling and Simulation*, 2nd edn, Academic Press, New York.