A TEMPORAL BLACKBOARD STRUCTURE FOR PROCESS CONTROL*

F. Barber, V. Botti, A. Crespo, D. Gallardo, and E. Onaindía

Grupo de Automática e Informática Industrial, Universidad Politécnica de Valencia, Apdo 22012, E-46071 Valencia, Spain

Abstract. Complex control systems require the use of new A.I. techniques to cope all the variable relations. One of the open topics is the use of temporal information and the ability of reason about it. Blackboard systems are one of the most promising proposal for real time expert system. In this paper a temporal blackboard to be used in a real time expert system is described. In order to represent the information a temporal model and its methods will be presented. Finally, some temporal features has been explained using a small scenario of a cement kiln process.

Keywords. Temporal reasoning, blackboard systems, real time expert system

INTRODUCTION

The temporal representation and reasoning problem arises in a wide range of Knowledge-Based System (KBS) application areas, where time plays a crucial role such as in process control and monitoring, fault detection, diagnosis and causal explanation, resource management and planning, etc. In these cases, temporal data representation is needed in order to obtain conclusions about the problem. Moreover, temporal dependence of knowledge relations (temporal constraints and parameterization of their application) is possible. However, neither the several Expert System (ES) prototypes developed to handle these applications [ISERMANN87, LAFFEY88, PERKINS90, TSANG88, etc.] nor the several expert system development tools which have a certain temporal capacity offer a general solution to the problem and the temporal representation and reasoning capacity of the ES has an **ad hoc** treatment in each case, which is too restrictive for the wide range of temporal domains. Temporal constraint and parameterization about rule application is not usually considered and the dynamic of the problem has not a natural representation.

In order to handle this problem in Artificial Intelligence (AI) areas (action-planning, qualitative reasoning about processes, explanation of the world at some earlier time, and prediction or temporal projection of the future) in a more specific way, several temporal logic-based models, from initial **situation calculus** [McCARTHY69], were defined [ALLEN84, McDERMOTT82, KOWALSKI86, SHOHAM88]. Thus, these temporal models in AI have an explicit time representation, based on time-points or intervals.

With a particular representation, the **Temporal Imagery** concept is related to representation and inference methods with large amounts of temporal information, and events and facts are arranged in a **time map**, based on temporal intervals [ALLEN83] or time points [DEAN87]. Other models are included in the Qualitative Reasoning concept [DeKLEER84, KUIPERS84, FORBUS84], with a qualitative consideration of the problem dynamic, about which the laws of change are defined.

However, some important points are not completely solved when temporal reasoning is used under time constraints. The need of manage past and future facts and the number of possible variable relations

*Partially supported by a grant of CICYT No. ROB89-0442 of Spanish Government and the ESPRIT Project REAKT 4651 (Thomson Syseca CRIN GMV UPV Marconi Etnoteam Computas)

can produce large operations for update and retrieve temporal information. Thus, an efficient management must be addressed.

In this paper a temporal blackboard (TBB) structure is described. The blackboard is a component in a real time expert system under development in the framework of the REAKT Esprit project [REAKT90]. In the following section, a revision of temporal information representation will be done. Next sections describe the temporal model assumed and the main methods for temporal management, and temporal functions provided by the blackboard to be used in rules. Finally, an example of a cement kiln subprocess will detail the use of the main functionalities.

TEMPORAL INFORMATION

Temporal information in control systems is relevant to know the evolution of the process and deduced variables and reason about it. The need of temporal information is related to past and future values. From a general point of view, a temporal value can be:

1. **exactly known**: when the time, where values were taken, is known (the alarm was 'on' from 10:20:03 till 10:28:45)
2. **partially known**: when a value is result a of sampling and this value is maintained till the next (the temperature at 10:23:22 was 120 degrees)
3. **known with an uncertainty**: when a variable will take a value in some instant of an interval (the kiln status will be HIGH before 20 minutes)
4. **dependent**: when the exact instant of a temporal value depend on other temporal facts (the alarm will be ON 20 minutes after the temperature be HIGH)

In terms of process control, 1 and 2 can be assumed as equivalent due to incoming values are produced at known instants (sampling period), and, if no more information is added, the same value can be maintained. This behavior if can be modeled by means of a *data persistence* as the time which the variable maintain its value. If this time is exceed a lack of information is produced and no value is assumed. When this persistence is ∞, the last value is always maintained.

Past values are always related to the instant they were produced, so, an absolute date can be used to store them. In this case there is not relation between temporal information, all of them are related to the clock and the relations must be deduced using clock values.

Future or prediction values introduce more difficulties due to the lack of knowledge about the exact instant they will be produced. In the third case, we predict that a status will be HIGH in some instant before 20 minutes, but we do not know the concrete instant it will start, finish or its duration. So, a model to represent and manage prediction has to represent this uncertainty and be able to reason about it.

Besides, fourth case introduces more complexity because of the dependency with another fact, when a fact occurs the dependent fact is equivalent to the third case, if it does not occurs neither the second one.

Moreover, the system behavior defines a set of temporal dependencies: a material in a cement kiln, from the input to the output takes about 10 minutes, so if we measure the temperature at the input, some prediction can be done about the future temperature in the middle and the output of the kiln, taking into account the process model, delays, etc. On the other hand, the system can take actions based in predictions in order to avoid problems in temperature.

In rule based systems, it means that rules should be fired with present and future facts and actions, in the right hand side of a rule, should be executed in the instant the values are deduced or when they are present.

TEMPORAL MODEL

The developed system is based on **time-points** (*tp*) as primitive for time representation. A *tp* represents a time instant in a discrete temporal line. An additional concept is the **temporal duration**(*td*) as the temporal distance between two *tp*'s.

The temporal relations relating two *tp*'s are:

- (BEFORE tp_i, td_k, [tp_j]): expresses that tp_i is temporally before tp_j plus a certain temporal duration td_k.
- (AT tp_i, td_k, [tp_j]): expresses that tp_i is temporally at tp_j plus a certain temporal duration td_k.
- (AFTER tp_i, td_k, [tp_j]): expresses that tp_i is temporally after tp_j plus a certain temporal duration td_k.

The td_k value, which stands for a temporal duration, may be positive or negative. The tp_j is optional and, by default, it represents the initial time reference (clock). With all these relations, the following temporal constraints can be set between time points:

- Temporal relation between a time point and the clock; the time point tp_i occurs at 8h:15':34", would be represented as:
  (AT tp_i (duration 8:15:34))

- Temporal distance between two time points; tp_i occurs at 10' AFTER tp_j is represented as: (AT tp_i (duration 0:10) tp_j)
- Relative constraints between two time points, when td_k=0; tp_i occurs before tp_j is represented as: (BEFORE tp_i (duration 0) tp_j).

These temporal relations are represented in a graph (**Time Map**) where nodes are the time points and the labeled directed edges show the temporal constraints between them:

Nodes: time points (tp_i)

Edges: Temporal constraints between time points: (**Before, At, After** td_k)

With this model as background, application objects are instances of classes with static and temporal attributes or slots, for instance:

```
Class Temperature has
  static attributes:
    name, location, units, ...
  temporal attributes:
    value
End class
```

Each object in the blackboard has a pointer to the parent class description and a set of static and temporal slots. The structure of a temporal slot is formed by the following pointers:

- object: which this slot belongs
- current: pointer to the temporal value that current now
- past: list of pointers to past temporal values
- future: list of pointers to future temporal values

The TEMPORAL_VALUE class is defined by means of a **value** and the **begin** and **end** points as the instant limits where the value was, is, or will be taken. Both time points are pointers to *tp* (instances of TM_NODE) in the **Time Map**. Each *tp* has a pointer to the slot which it belongs, and three lists of *tp*'s with relation of BEFORE, AFTER, and AT associated with a temporal distance.

The following figure 1 shows the relations between these data structures.

TEMPORAL MANAGEMENT

The set of management routines of a Time Map is called **Time Map Manager** (TMM). TMM functions manage the basic operations of creation, deletion, and temporal relations management between temporal values. In order to manage future values

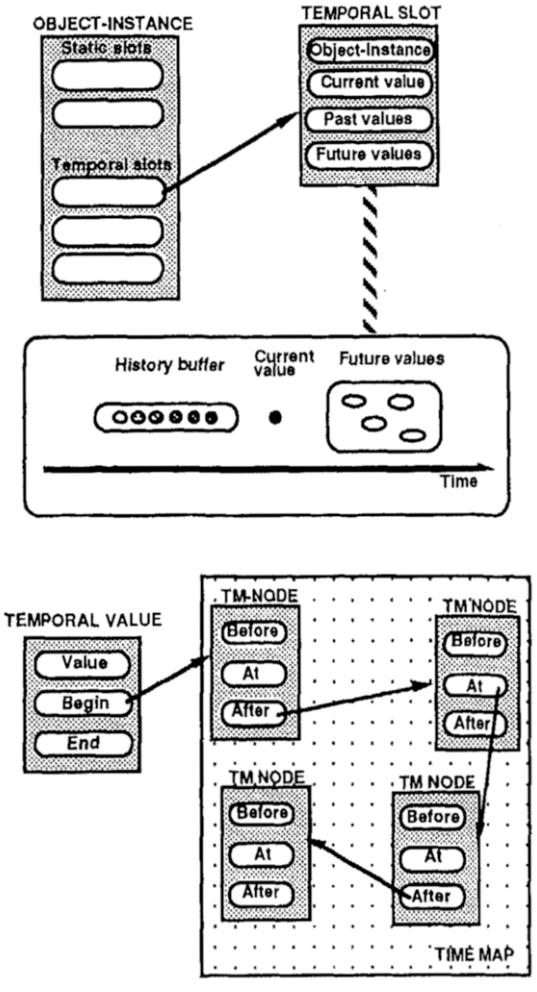

Figure 1: Temporal data structures

in a proper way, a suitable solution is to provide a reason maintenance system (RMS) cooperating with the TMM. Both components allow to manage logical dependencies and coherence of temporal values.

The main operations of a TMM are the updating and retrieving temporal information. Following sections describe these processes in depth.

Updating process

The function for updating a temporal constraint between two time points with relation **rel** (BEFORE, AFTER, or AT) and a distance is:

addTemporalRelation(tp_i, *rel*, td_k, tp_j)

The updating process of a new constraint performs a consistency and non-redundancy checks. If the new constraint is consistent and non-redundant with the existing ones, the new constraint is asserted in the Time Map. Thus, the result of this operation may

be:

- Contradictory, whether the temporal constraint to be updated is inconsistent with the information already represented in the TBB. In this case the TBB keeps with no modifications.
- Redundant, whether the temporal constraint to be updated can be retrieved from the ones existing in TBB, and so it will not be introduced.
- Updated, whether the temporal constraint is updated.

Retrieval process

Two basic retrieval operations are performed by the Time Map Manager: retrieve a concrete temporal constraint between two time points and evaluate the more constrained temporal distance between two time points.

(retrieveTemporalRelation (tp_i, AFTER, td_k, tp_j)

(retrieveTemporalDuration (tp_i, tp_j)

The retrieval of an specific temporal constraint between two nodes is stated as a path searching process between the two nodes throughout the rest of nodes in the Time Map. Due to the existence of many possible paths between the two nodes, the goal is to find a restrictive path according to the inquired constraint by combining the temporal constraints of the edges in the path. That is, to obtain a path which permits to provide an answer 'true' or 'false', if it is feasible to be deduced from the Time Map. The result of this function may be: **True, False, or Possible.** A response Possible can be obtained in two cases: i) There is no path relating both nodes, or ii) Though existing a path, the information provided is not enough to deduce neither the constraint nor the opposite one.

In this way, the stated retrieve operation consists of obtaining the shortest combined path between two time points in a directed graph whose edges are labeled with a temporal relation and a weight (positive or negative). This algorithm is based in a bidirectional search from each time point involved in the expression looking for the direct and inverse relation in the graph.

The second recovery processes obtains a response about a concrete temporal constraint between two time points. In order to evaluate the more constrained temporal distance between two time points the same process can be used. Thus, the exact temporal distance between the two *tp*'s, or, the maximum and minimum temporal distance between *tp*'s can be obtained.

Evaluation

Above operations are the most complex in the temporal information management. They can decide whenever a model is appropriated for real time systems or not. The described temporal model processes has been implemented by means of efficient algorithms [BARBER92] in Common-Lisp running on a Sun SparcStation 2 and tested by running a great number of representative empirical proofs on Time Maps randomly generated. Figure 2 shows the maximum time for each TMM process for a Time Map with 30% of nodes related with the clock and, four temporal constraints for each node as average ratio.

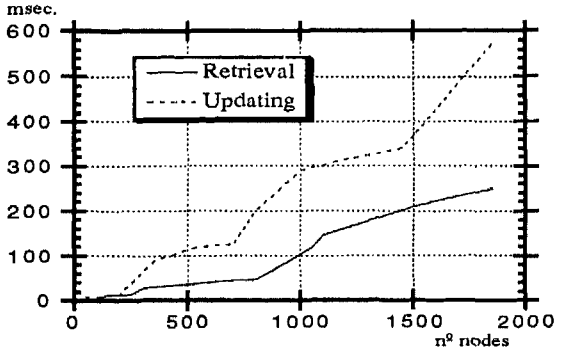

Figure 2: Updating and retrieving processes evaluation

In this Time Map, 1800 nodes has been introduced with 600 numerical constraints between a node with the clock and 7200 between two nodes. Computational times have been measured each 10 nodes. Taking into account the test results, a nearly linear behavior for the updating and retrieving processes can be assumed. The computational cost for the deletion process is constant (10 msec). Moreover, the spatial cost in all cases was not relevant.

BLACKBOARD STRUCTURE

Blackboard structure provides storage and management of application objects used by concurrent reasoning processes. Figure 3 shows the relations between all components in a real time expert system.

Blackboard interface provides a set of functions that can be used by a user language to express actions, such as class definition, object instantiation, and temporal updates and queries. Some of these functions can be used as rule predicates in the left hand side, and others, called in the right hand side, that updates the internal information.

With respect to the **Left Hand Side** temporal functions provided by the blackboard, they can be stated as:

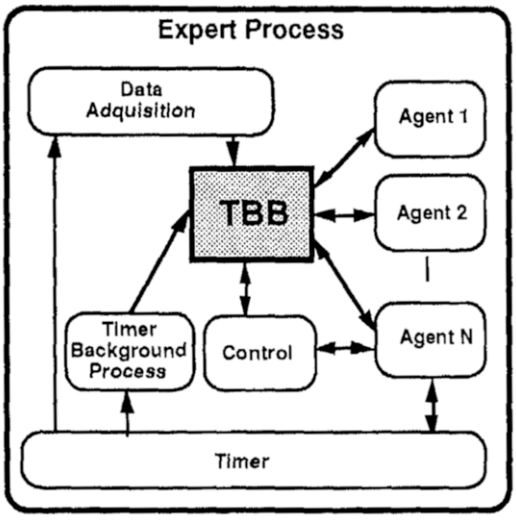

Figure 3: Blackboard relations

- when_occurs (object slot begin end)
  This predicate is satisfied for those temporal values (current or predicted) which, belonging to the temporal slot, match the value. The begin and end of the temporal value may be instantiated in variables. They can be either an absolute value or a *tp*.

- past_value (object slot value begin end)
  This predicate is satisfied when a current value in the defined slot is outdated and is moved from current to past.

- temporal_test (any temporal expression)
  Time-test function is used for asking about i) temporal relations between dates or begin or end of temporal values or ii) between a date or time points obtained from the begin or end of a temporal value and the current time (NOW).

  The temporal_test management is achieved by using the TMM internal functions to to inform in the moment that a time point value changes its temporal window.

Blackboard functions in the **Right Hand Side** of the rule modify the internal state by adding values and predictions about slot values of nodes.

- put_value (object slot value begin end dependencies)

  This function adds a value to a slot. If the slot is a temporal slot, then the function is used to create a current or a predicted temporal value according to the premises of the rule:

  - if there is at least one premise instantiated to a predicted temporal value, then the asserted value will be predicted
  - if all the premises are instantiated to current temporal values then the asserted value will be current.

  In the former case, when all the predicted premises supporting the asserted value change to current, then the created value also change to current. In this case, the existing temporal constraints for its end-time are retracted and only the temporal constraint (after begin) is maintained. And if some predicted premise is not fulfilled, then the asserted predicted value will be deleted.

  When a current value is asserted in a slot, if there is a previous current value in it, then this current value is moved to its historical buffer of past values. The end time of this previous current value is modified with the date corresponding to the begin of the new created current value.

- predict_value (object slot value begin end dependencies)

  Adds a predicted value to a temporal slot. The user specifies the begin and end date for the value, by writing temporal restrictions expressing BEFORE, AFTER or AT a time point.

  The status of the created value is always predicted. It may change according to the following specification

  - to current: a predicted value created with this function is changed to current when the prediction is fulfilled, this only can occurs when a new value (that matches with the prediction) arrives by means of a put_value.
  - deleted: a predicted value is deleted in two cases: i) when arrives the upper time of its begin temporal window and the prediction has not been fulfilled and ii) when was created with predictions in the LHS of the rule and one of this predictions is not fulfilled.

The following meaning for the predicted values is assumed: a prediction is **matched** by a new (predicted) temporal value when it has the same value and the begin and the end of the new temporal value are respectively within the temporal windows of the begin and end of the prediction. Then the temporal window of the matched prediction can be constrained according the new prediction. In this case, pending queries can be updated. A prediction is **fulfilled** by a new (current) temporal value when it has the same value and the begin and the end of the new temporal value are respectively within the temporal windows of the begin and end of the prediction.

If there is some prediction about the slot which is fulfilled by the put_value (current value), then it is not created a new current temporal value. Then the status of the fulfilled prediction value is changed to current. In this case, the temporal constraints for its

end time are deleted, since a current value has not end temporal constraints. Only the temporal constraint (after begin) is maintained.

Blackboard informs to other components in the system when new current, past or prediction values are created or predictions fulfilled or deleted.

EXAMPLE

In order to illustrate the main features of the proposed functionalities a small example is described. The example is based in the temperature subprocess of a cement kiln.

There are three **temperature sensors** at beginning, middle and end of the kiln. From these inputs, some internal variables (**kiln status** as hot normal or cold) can be deduced. There are three variables that permit control the kiln status at beginning, middle and end: the input of fuel, flow of fumes, and external air flow, respectively. There is a delay in the raw material flow along the kiln.

The system can be modeled by means of three classes: SENSOR, VALVE, and STATUS, with some static attributes as *name*, *location*, *range*, etc., and a temporal attribute *value*. Another class KILN models the relations between all objects. There are defined three temperature sensors TS1, TS2 and TS3, three kiln status KS1, KS2, and KS3, and three valves V1, V2, and V3. A persistence is assumed for the sensor inputs.

The following rules express the system behavior:

R1: Determines the value of the status from its sensor.
when a new sensor data is added,
if the temperature value is high,
then the status is hot.

Some alternative **R1** rules deduced the normal and cold status.

R2: Predicts a change in the next temperature point taking into account the flow delay. A minimum *dmin* and maximum *dmax* delay is assumed.
When a new status is added,
if the kiln status is hot
then predict high for the next temperature value with a begin point after(dmin) and before(dmax).

R3: Takes decision about open or closed the associated valves to a determined point.
When a new status is added,
if the kiln status is hot (cold)
then closed (open) now the associated valve.

Let consider the following situation: at time T1 a new sensor data TS1 arrives, and the reasoning process is performed at T2 producing the results showed in figure 4.

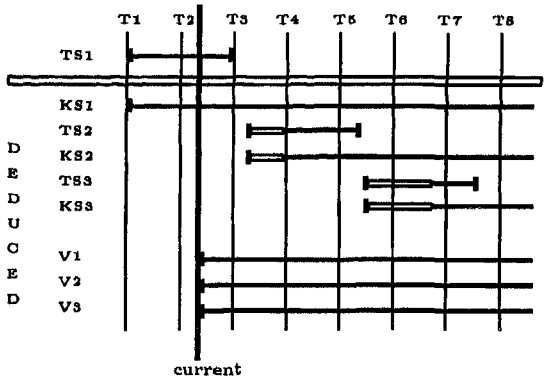

Figure 4: Initial deductions

KS1 is deduced as hot in [T1, ∞], TS2 is predicted as high with a delay and as consequence KS2 hot. As a prediction on KS2 is produced rule R3 is instantiated with this new fact and another prediction on TS3 and consequently on KS3 is produced. Each time a value for some KS is added an action of open or close the associated valve is produced but in the current time T2.

In this scenario if a time T3+D a new value of TS2 is read matching the prediction, the following situation is obtained (figure 5).

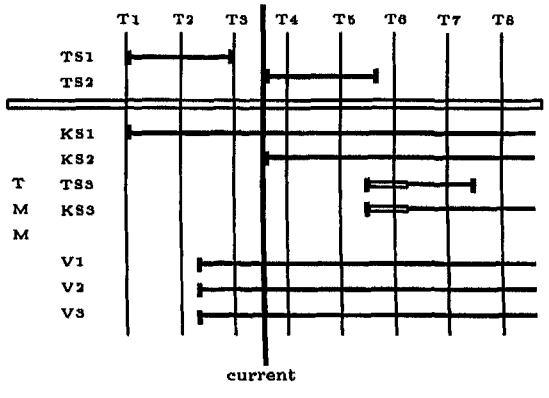

Figure 5: Prediction matched

Note that the temporal window of TS3 and KS3 are updated being more restricted due to the new added knowledge.

Now, let consider when a prediction is not matched. If any TS2 match the prediction, at time T4 (end of begin window) predictions about TS2, KS2, and obviously TS3 and KS3 are removed (figure 6).

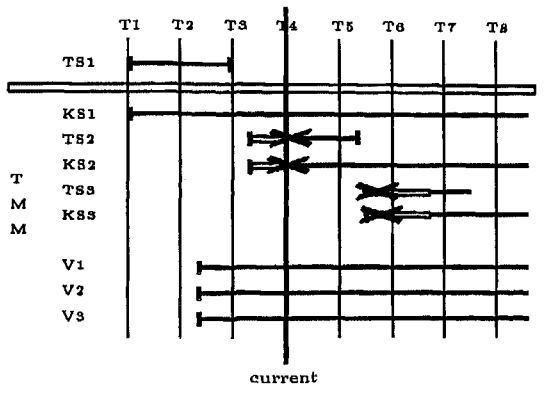

Figure 6: Prediction not matched

CONCLUSIONS

In this paper, a temporal blackboard has been presented. The temporal information is associated to the temporal attributes of objects stored using a model based on time points. The proposed model permits to represent a great number of situations including uncertainty. In order to have efficient methods to be applied under time constraints, past facts are stored with respect an absolute clock and future using a graph when there is no possible due to the facts dependencies. Fast algorithms permits to update and retrieve temporal information in the Time Map. Some results of these operation has been showed.

Finally, some temporal features has been explained using a small scenario of a cement kiln process.

Acknowledges

The authors acknowledge to T. Chehire, A. Mensch, J.Y. Quemeneur, D. Kersual, J.J. Galán, I. Rodríguez, E. Pezzi, C. Luparia, A. Lupi, M. Shenton and R. Fjellheim its comments and its participation in the global architecture described in this paper.

REFERENCES

[ALLEN83] J.F. Allen "Maintaining knowledge about temporal intervals". *Comm. of the ACM.* Vol 26, No. 11, 1983.

[ALLEN84] J.F. Allen "Towards a general Theory of Action and Time". *Artificial Intelligence*, 23, 1984.

[BARBER92] F. Barber, E. Onaindía, M. Alonso. "Representation and management of temporal relations". *Novática.* 1992.

[DEAN87] T. Dean and D. McDermott "Temporal Data Base Management" *Articial Intelligence*, 32, 1987

[DeKLEER84] J. De Kleer, J.S. Brown. "A Qualitative physics based on confluences." *Articial Intelligence*, 24 August (1984), pp.7-83.

[FORBUS84] K.D. Forbus. "Qualitative process theory" *Articial Intelligence*, 24 August (1984) pp.85-168.

[ISERMAN87] R.Isermann (Ed.). *Procc. of the 10th world congress on automatic control.* IFAC-87, vol.6, (1987).

[KOWALSKI85] R.A.Kowalski, M.Sergot. "A logic-based calculus of events." *New Generation Computing*,4 (1).(1985).pp.67-95.

[KUIPERS84] B.Kuipers. "Commonsense reasoning about causality: Deriving behavior from structure" *Artificial Intelligence*, 24, pp.169-203.

[LAFFEY88] T.J. Laffey, P.A. Cox, J.L. Schmidt, S.M. Kao and J.Y. Read *Real-Time Knowledge-Based Systems. AI Magazine.* Spring. 1988.

[McCARTHY69] J.M. McCarthy, P.J. Hayes. "Some philosophical problems from the standpoint of artificial intelligence." *Machine Intelligence*,4. Edinburgh Univ. Press. (1969).

[McDERMOTT82] D.V. McDermott. "A temporal logic for reasoning about processes and plans." *Cognitive Science*,6. (1982). pp.101-155.

[PERKINS90] W.Perkins, A.Austin. "Adding temporal reasoning to expert-systems building environments". *IEEE-Expert.* Freb.1990.

[REAKT90] Thomson, Syseca, Crin, GMV, UPV, Marconi, Etnoteam, Computas. *REAKT: Environment and Methodology for Real-time Knowledge Based Systems.* ESPRIT II 4651. 1990-93

[SOHAM88] Y. Shoham. *Reasoning about change.* MIT Press.(1988)

[TSANG88] E.Tsang. "Elements in temporal reasoning in planning" *Procc. of ECAI-88 Munchen.* pp.571-573.