Sixth Int. Conf. on IFSs, Varna, 13-14 Sept. 2002
NIFS 8 (2002), 3, 57-65

Representing and Querying Temporal Workflow Schemas

Hathal Al-Roki[1], Panagiotis Chountas[2], Ilias Petrounias[1],
Vassilis Kodogiannis[2], Boyan Kolev[3]

[1] -Department of Computation UMIST, Manchester PO BOX 88 M60 1QD, UK
[2] - Mechatronics Group, Dept. of Computer Science, Univ. of Westminster, London, HA1 3TP, UK, e-mail: chountp@wmin.ac.uk
[3] - Centre for Biomedical Engineering - Bulgarian Academy of Sciences, Acad.G.Bonchev Str., Bl.105, Sofia-1113, Bulgaria, e-mail: bobby_kolev@yahoo.co.uk

Abstract

Workflows are activities involving the co-ordinated execution of multiple tasks performed by entities, in the enterprise environment. Workflow systems support the encoding and execution of workflows. A workflow management system is proposed, in which a considerable attention is paid on the utilisation of the services provided by the underlying flexible database formalism. Flexible database formalisms must permit the modelling of trends, seasonality (periodicity), cyclic variations of trends and irregular activities (no predictable patterns) in the enterprise environment This paper will be restricted in workflow management system design, and activity scheduling, in a post relational environment whereas relational operators, utilise paths (links between domains), in order to express interdependencies between activities.

Keywords: Temporal Activities, Time Model, Temporal Activities, and Temporal Workflows

1. INTRODUCTION

Workflow Management systems improve business process by integrating information, from more than one domains, directly or indirectly related with a specific enterprise function. In a non-stable and changing business environment, there is a critical need to become more competitive, by controlling the flow of information throughout the enterprise in a timely manner.

Many business activities have restrictions such as a constrained duration, dates of resubmission, maximum and minimum duration of an activity. Typically time violations increase the cost of a business function, because they introduce some kind of exception handling [1]. Existing workflow management systems offer limited support for management and representation of activities with constrained or infinite duration. The latter includes periodical and recurring activities [2] with either known or known unknown frequency of reoccurrence. Therefore a workflow management system should provide information about an activity, its calendric restrictions, and the time requirements about the activity.

This paper presents a temporal post relational environment that utilises the concept of lattices and hierarchies for the support of the major components defined within a workflow environment. In representing activities, encoding seasonality and trends a temporal model and representation at the database level is required and thus is proposed. The way that parallel and conditionally executed activities are related through the dimension of time, is defined and presented.

The rest of the paper is organised as follows. Section 2 defines a reference model for workflow management. Section 3 defines the basic elements of temporal representation. Section 4 introduces an object role formalism for encoding activity states. Section 5 specifies the notion of activity and state relations. Section 6 captures and query, activities and their paths as part of a post relational environment. Section 7 concludes and points to future extensions of the current framework.

2. REFERENCE MODEL FOR WORKFLOW MANAGEMENT

The Workflow Management coalition [3] defines a reference model, which describes the major components of workflow architecture. According to this reference model for major elements can be identified in describing the paths between activities participating in a workflow schema:

Sequence: Activities are executed in sequence (e.g. one activity is executed after another)

Parallel: Two or more activities are executed in parallel. Two different blocks can be identified: AND-Split, AND-Join. Here synchronisation is required. *a)* The AND-Split enables parallel execution of two or more activities after successful completion of the parent-activity. *b)* The AND-Join enables synchronisation of two or more parallel activities into a inheritor activity after successful completion of the ancestor activities

Condition: one of the selected activities is executed. Modelling a choice among two or more selections two blocks may be utilised: a) XOR-split and b) XOR-join. Here no synchronisation is required.

Iteration: sometimes it is crucial to execute a single activity or a set of activities repeatedly over time. The novelty in our approach is the capture and representation of the iteration feature as an intrinsic part of a time model.

Next for case study purposes in this paper a sample workflow schema is introduced, which presents some of the above features.

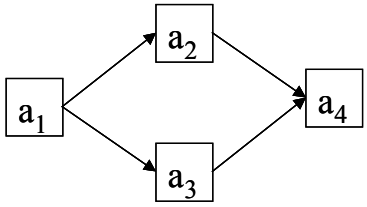

Figure 1. W, Sample Workflow Schema

In the above schema (Fig 1) the following activities can be identified (a_1, a_2, a_3, a_4). Activities (a_2, a_3) are the AND-Split, XOR-Split products of (a_1). Similarly for explanatory purposes it can be assumed that (a_4) is the AND-Join or the XOR-Join.

Each separate activity (e.g.(a_1)) is temporally constrained with the aid of temporal inequalities. One can differentiate between local and general inequalities [4]. Using both types of temporal constraints the enforcement of strict or reluctant terms determined by the organisation hierarchy and have to be satisfied by a specific activity. For example considering activity (a_1) a strict situation is determined by a local constraint as follows:

Strict situation (S_1): $G_1 \leq \Delta T(S_1) \leq G_2$, (1) where G_1 G_2 are defined with precision on top of a linear time hierarchy, in that sense a calendar may be arbitrary [5]. A strict situation simply declares that time constraints implied to a specific activity must be met with precision.

Reluctant situation (S_2): $G_1 \leq \Delta T(S_2) \leq G_2$, (2) and G_1 ,G_2 are further constrained with the aid of local inequalities $C_L \leq G_1 \leq C_R$, $C_{L1} \leq G_2 \leq C_{R1}$, where G_1, G_2, are defined with imprecision on top of a linear time hierarchy or an arbitrary calendar. C_L, C_R, C_{L1}, C_{R1} are defined with precision of a linear time hierarchy. A reluctant situation simply declares that the duration of an activity cannot be

specified exactly. Using the best case approach it is expected that an activity will start at the earliest time point C_L and it will finish at earliest time point C_{L1}. Using the worst case approach it is expected that an activity will start at the latest time point C_R and it will finish at latest time point C_{R1}.

Next a temporal formalism is presented for the encoding of activity instances with definite, or constrained duration, that may be repeated over the time dimension.

3. TIME MODEL

In this section the basic elements for a temporal representation are defined. The central concepts are a timeline and a time point where the former is comprised of the latter. A timeline is an ordered continuous infinite sequence of time points, whereas a time point is a particular instantaneous point in time. The term duration is defined as an absolute distance between two time points or it may also imply the existence of two bounds an upper bound and a lower bound (indefinite temporal Information). The concepts of duration (D) and time interval are defined with the help of time points. A time interval is defined as a temporal constraint over a linear hierarchy of time units denoted H_r. H_r is a finite collection of distinct time units, with linear order among those units. For instance, H_1=day⊆month⊆year, are all linear hierarchies of time units defined over the Gregorian calendar. If we are representing the Gregorian calendar by using hierarchy H_1 from above, a suitable validity predicate states that, for example, valid (14/9/1995) = true but valid (29/2/1998) = false.

A time interval is presented in the form of $[C+K\times X, C'+K\times X]$ where $C'=C+D$, $D\in N^*$, thus an interval is described as a set of two linear equations defined in a linear time hierarchy (e.g. H_2 = day⊆month⊆year).

The lower time point $t_{Earliest}$ is described by the equation $t_{Earliest} = C+K\times X$. The upper point t_{Latest} is described by the equation $t_{Latest} = C'+K\times X$. C is the time point related to an instantaneous event that triggered an activity, K is the repetition factor, $K\in N^*$ or the lexical 'every' (infinite-periodical information).X is a random variable, $X\in N$, including zero, corresponding to the first occurrence of an activity instance restricted by a constraint. The product $K\times X$ is defined according to a linear hierarchy. D represents the duration of an activity instance, how long it is valid for after it has been introduced, D is also mapped in a linear time hierarchy and may be in the range between a lower and upper bound $G_L \le D \le G_R$ Constraints are built from arbitrary linear equalities or inequalities (e.g. $t_{Earliest}=C+7X$ and $0\le X\le 5$). Limiting the random variable X results in specifying the lower and upper bound of a time window.

When K=0 *Definite or Indefinite* temporal information can be represented. In the case of *Indefinite* temporal information the following constraints about the duration of an activity instance are applicable e.g. $G_1 \le D \le G_2$.

When $0\le X\le$n, x$\in N^*$ and $K>0$, then *Infinite Temporal Information* is represented. The duration D of an activity instance can be either absolute or bounded

Infinite Temporal Information: is defined when an infinite number of times are associated with an activity instance [2], [4]. To be able to present this kind of information effectively, a finite representation is needed. Infinite temporal information includes the following types of information:

a) Periodic: An activity instance is repeated over a time hierarchy with the following characteristics: a constant frequency of repetition K, it has an absolute and constant duration D, and X a random variable that denotes the number of reappearance's for an activity instance. Therefore the duration of every activity instance constituting an activity type and consequently the duration of an activity type is well known.

b) Unknown Recurring: Generally is described in the following intervalic form, t = [⊥, ⊥]. The intuition is that the duration D of an activity instance is assumed to be known or constrained

and the frequency of reoccurrence ($K=?$) is not known. However by definition it is known that if an activity instance is recurring, then its next reappearance cannot occur before the previous one is ended. It is also known that $K \geq 1$. Therefore the following conclusion can be made $D \leq K$.

Next, an activity model is presented. The model defines an activity in terms of the structural elements of a specific Universe of Discourse (UoD).

4. TIME TEMPORAL ACTIVITY MODEL (TAM)

This section introduces an object-role based formalism for representing activities and their instances (states) defined over time. The model defines an activity in terms of the structural elements known as static. However, in the context of this paper the term structural is more appropriate because evolution of an activity through its states, can also be captured and thus the term "static" seems restrictive.

A temporal workflow environment is containing information about the past and present of the modelled world. The approach followed here is based on a type of object role modelling formalism. In that sense an activity is a true logical proposition about the modelled world. Each activity instance or state is a semantically non-deductible proposition in the real world about one or more entity instances. Non-deductible means that the activity type cannot be split into further activities involving fewer entities without loss of information. The basic items that one wishes to reason about are objects in terms of the roles that they play within a domain [6]. An activity type in TAM is composed of the arguments shown in Fig. 2, where n is the arity of the activity type. The way that one can refer to specific entities is through reference labels.

$a_k = \{\{<E_1, L_1, R_1> \ldots <E_n, L_n, R_n>\}, \{ G_L \leq \Delta T(a_{k1} \ldots a_{kv}) \leq G_R \}\}$

Where:
a_k is a activity type consisting of k activity instances
E_i is the entity type playing a role in the activity type
L_i is the label type (referencing E_i)
R_i is the role of the activity type
Δt is the constrained time interval that an non-deductible activity type is defined in the real world

Figure 2. Activity Types

A graphical representation of the concepts is shown in Figure 3

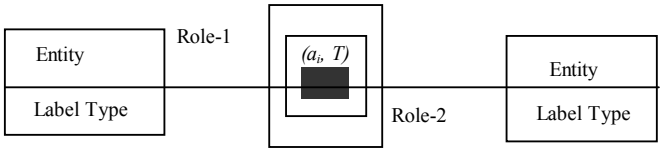

Figure 3. Graphical Notation of an Activity Type

The proposed structural formalism supports the states (activity instances) of a particular activity over an arbitrary linear time hierarchy. Two different types of relations are introduced. One type is noted as an activity relation, and the other as a state relation. An activity relation in accordance to the workflow reference model represents the three major elements (*sequence, parallel, condition*) that can be identified in describing the paths between activities participating in a workflow schema. A state relation is defined as the evolution of an activity through the time and is a direct mapping of the activity model (TAM). A state relation encodes the time dimension,

permitting the expression of definite, constrained and recurring activity states, capturing thus the *iteration* factor, as part of the temporal, dimension.

5. DEFINING STATE AND ACTIVITY RELATIONS

In this section the two types of relations that will be used for modelling paths between activities and activity sates are formally defined. The former is known as an activity relation, while the latter is defined as a state relation.
In order to represent a workflow schema W (Figure 1) hierarchical structures are used instead of flat tables [7], [8]. A relation schema R is recursively defined as:

If $\{A_1, \ldots, A_n\} \subset U$ and $A_1 \ldots An$ are atomic valued or zero order attributes then $R=\{A_1,..., A_n\}$ is a relation schema.

If $\{A_1, \ldots, A_n\} \subset U$ and $A_1 \ldots A_n$ are atomic valued attributes and $R_1,\ldots,R_n$ are relation schemas or high order attributes then $R=(A_1,\ldots,A_n,R_1 \ldots R_n)$ is a relation schema.

An activity relation R_A: a relational schema of high and atomic order attributes, in which a high order attribute corresponds to a path between activities, (*sequence, parallel, condition*), thus constituting the workflow schema W (a_1, a_2, a_3, a_4) in its entirety (Fig 1).

A state relation: R_S in turn shows the different states (or activity instances) consisting of an individual activity (a_1) which participates in an activity relation R_A. The activity identifier indicates a link between a state and activity relation. Formally a state relation is defined as follows;

Definition: Let T a set of time intervals $T=\{t_L, t_R\}$ where $t_L=C+K\times X$, $t_R=C'+K\times X \wedge a_1 \leq X \leq a_v\}$ and D a set of non temporal values. A generalised tuple of temporal arity x and data arity l is an element of $T^x \times D^l$ together with constraints on the temporal elements. In that sense a tuple can be viewed as defining a potentially infinite set of tuples.

Considering that activity relations refer to state relations there is a need to define the relationship in terms of time between the two different type of relations. Furthermore activity instances can have their own evolution over the time but always between the time limits specified in either *Strict situation (S_1)* or *Reluctant situation (S_2)* (section 2). In that sense if a reluctant or strict situation are not predetermined, we can still derived a temporal inequality for a specific activity (e.g. a_1), which is restricted between the earliest time point

$$G_L = [G_1 = t_{\text{Best-Earliest}}(a_{1\iota} \ldots a_{1v}),\ G_1' = t_{\text{Worst-Earliest}}(a_{1\iota} \ldots a_{1v})]$$

& the latest time point,

$$G_R = [G_2 = t_{\text{Best-Latest}}(a_{1\iota} \ldots a_{1v}),\ G_2' = t_{\text{Worst-Latest}}(a_{1\iota} \ldots a_{1v})]$$

noted as follows

$$G_L \leq \Delta T(a_{1\iota} \ldots a_{1v}) \leq G_R \quad (3)$$

$$t_L a_{1\iota} \leq \Delta T a_{1\iota} \leq t_R a_{1\iota}$$
$$\ldots.$$
$$t_L a_{1v} \leq \Delta T a_{1v} \leq t_R a_{1v}$$

$$G_1 = t_{\text{Best-Earliest}}(a_{1\iota} \ldots a_{1v}) = \min(t_L a_{1\iota} \ldots t_L a_{1v}) \quad (4)$$
$$G_1' = t_{\text{Worst-Earliest}}(a_{1\iota} \ldots a_{1v}) = \max(t_L a_{1\iota} \ldots t_L a_{1v}) \quad (5)$$
$$G_2 = t_{\text{Best-Latest}}(a_{1\iota} \ldots a_{1v}) = \min(t_R a_{1\iota} \ldots t_R a_{1v}) \quad (6)$$
$$G_2' = t_{\text{Worst-Latest}}(a_{1\iota} \ldots a_{1v}) = \max(t_R a_{1\iota} \ldots t_R a_{1v}) \quad (7)$$

Where $a_{1\iota}$ is the first activity instance a_{1v} is the latest activity instance (state). $(a_{1\iota} \ldots a_{1v})$ is the set of the activity instances (states). Therefore (4), (5), (6), (7) are defining the best and worst earliest or latest times that a state relation R_S is defined over time.

In the general case (4), (5), (6), (7) can be used to define the best and worst earliest or latest times that an activity relation R_A is defined over time. However in the case of parallel execution synchronisation is required. Additional or spare time t_s is needed for synchronising parallel activities. Synchronisation is required only in the case of the AND-Join. Synchronisation will occur between the latest time points between activities under AND-Join.

In Fig1, (a_2, a_3) are executed in parallel. Assuming that a_2, finishes before a_3 then a_2 have to wait until a_3 is completed. Thus (6), (7) are modified as follows;

$$G_2 = t_{Best\text{-}Latest}\ (a_{1t}\ldots\ a_{1v}) = \min(t_R a_{1t}\ldots\ t_R a_{1v})\ + t_s \quad (8)$$

$$G_2' = t_{Worst\text{-}Latest}(a_{1t}\ldots\ a_{1v}) = \max(t_R a_{1t}\ldots\ t_R a_{1v}) + t_s \quad (9)$$

In summarising it can be said that the distinction between activity and state relation, permits the separable expression of paths between activities in a workflow schema, with the aid of an activity relation, while a state relation allows the independent evolution of an individual activity through its states over the time.In the following section the workflow schema presented in Fig 1 is translated to a temporal database representation with the aid of activity and state relations. Relational algebraic operations are defined for the expression of the major components of a workflow architecture (*sequence, parallel, condition*).

6. REPRESENTING WORKFLOW SCHEMAS

In this section the activity relations for the AND-Split, AND-join, XOR-Split, XOR-Join are defined and then it is shown with the aid of an extended relational algebra how the workflow schema (Fig 4) in its entirety can be derived.

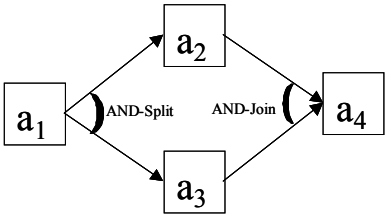

Figure 4. W, Sample Workflow Schema with Paths

With reference to Fig 4 the following activity relations are defined $R_{AND\text{-}Split}$ and $R_{AND\text{-}Join}$. For each activity relation and as a result for each join operator a new time dimension of time is introduced named as path time **PT(R_A)**. The $R_{AND\text{-}Split}$ activity relation is defined as follows

Table 1. **$R_{AND\text{-}Split}$** Activity Relation

$R_{AND\text{-}Split}$	**Act-1**	**Act-2**	**PT($R_{AND\text{-}Split}$)**
X_1	a_1	a_2	$G_{L1} \le \Delta T(a_{1t}\ldots a_{1v}) \le G_{R1}$ $\cap$ $G_{L2} \le \Delta T(a_{2t}\ldots a_{2v}) \le G_{R2}$
X_2	A_1	a_3	$G_{L1} \le \Delta T(a_{1t}\ldots a_{1v}) \le G_{R1}$ $\cap$ $G_{L3} \le \Delta T(a_{3t}\ldots a_{3v}) \le G_{R3}$

G_{L1} , $G_{L2,}$ G_{L3} are defined individually by (4) and (5). Similarly G_{R1} , $G_{R2,}$ G_{R3} are defined by (6) and (7). The semantics of the AND-Split are determined that $G_{R1}=G_{L2,}$ and $G_{R1}=G_{L3}$. A path time **PT(R_A)** is determined using the intersection semantics and is defined as follows

$PT(R_{AND\text{-}Split}) = VT(R(a_1)) \cap \ldots \cap VT(R(a_v))$ where $VT(R(a_1))$ the valid time that a_1 is defined through its activity states presented by a state relation.

e.g. $PT(R_{AND\text{-}Split}) = VT(R(a_1)) \cap VT(R(a_2)) = [G_{R1}, G_{L2}] = [G_{R1}, G_{R1,}]$ (10)

$PT(R_{AND\text{-}Split}) = VT(R(a_1)) \cap VT(R(a_3)) = [G_{R1}, G_{L3}] = [G_{R1}, G_{R1}]$, (11) is a dummy temporal interval since the upper and lower ends are equal.

In the same way the $R_{AND\text{-}Join}$ activity relation can be defined. G_{L2} , $G_{L3,}$ G_{L4} are defined individually by (4) and (5). G_{R4} is defined by either (6) or (7). Similarly G_{R2} , $G_{R3,}$ are defined by (8) and (9). Equations (8) and (9) are achieving synchronisation between activities a_2 and $a_{3.}$ The semantics of the AND-Join are determined as $G_{R2} + t_s = G_{L4,}$ and $G_{R3}=G_{L4}$ where t_s is the time needed in achieving synchronisation between $a_{2,}$ and $a_{3.}$ Assuming for example that a_2, finishes before a_3 then a_2 have to wait until a_3 is completed.

Table 2. $\mathbf{R}_{AND\text{-}Join}$ Activity Relation

$\mathbf{R_{AND\text{-}Join}}$	Act-2	Act-3	$\mathbf{PT(R_{AND\text{-}Join})}$
X_1	a_2	A_4	$[G_{L2} \le \Delta T(a_{2_1} \ldots a_{2_v}) \le G_{R2} + t_s]$ $\cap$ $[G_{L4} \le \Delta T(a_{4_1} \ldots a_{4_v}) \le G_{R4}]$
X_2	A_3	A_4	$[G_{L3} \le \Delta T(a_{3_1} \ldots a_{3_v}) \le G_{R3}]$ $\cap$ $[G_{L4} \le \Delta T(a_{4_1} \ldots a_{4_v}) \le G_{R4}]$

Thus $PT(R_{AND\text{-}Join}) = VT(R(a_2)) \cap VT(R(a_4)) = [G_{R2}+t_s, G_{L4}] = [G_{L4}, G_{L4}]$ (12)

$PT(R_{AND\text{-}Join}) = VT(R(a_3)) \cap VT(R(a_4)) = [G_{R3}, G_{L4}] = [G_{L4}, G_{L4}]$ (13)

Considering (10), (11) then the activity relation $R_{AND\text{-}Split}$ can be restructured as follows (14), presented in Figure 5

$$R_{AND\text{-}Split} = [(a_1 (a_{2,} a_3))\ PT(R_{AND\text{-}Split})] \quad (14)$$

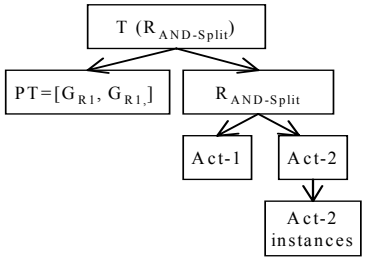

Figure 5. $R_{AND\text{-}SPLIT}$

Considering (12), (13) then the activity relation $R_{AND\text{-}Join}$ can be restructured as follows (15), prented in Figure 6

$$R_{AND\text{-}Join} = [((a_{2,} a_3)\ a_4)\ PT(R_{AND\text{-}Join})] \quad (15)$$

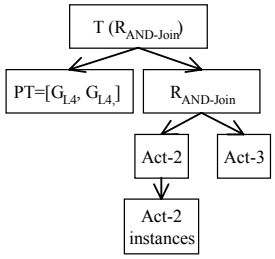

Figure 6. $R_{AND\text{-}JOIN}$

Joining (14), (15) then the workflow schema (W), (16) is derived and presented respectively through R_W in Figure 7

$$R_W = [(a_1 (a_{2,} a_3)\ a_4)\ PT(R_w)] \ (16)$$

Where $PT(R_W) = \min [PT(R_{AND\text{-}Split}), PT(R_{AND\text{-}Join})] \,\&\, \max [PT(R_{AND\text{-}Split}), PT(R_{AND\text{-}Join})] = [G_{R1}, G_{L4}]$

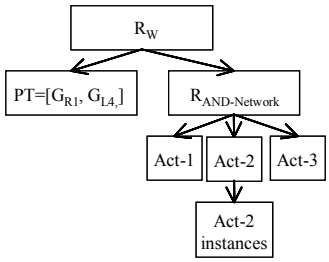

Figure 7. Workflow schema W, through activity relation R_W

Selection (σ): For all nodes $\in$ node S where $S_a \neq S_b$, if node S_a is a child of an ancestor of a node S_b, then S_a, S_b are called selection comparable nodes ($S_a \overset{\sigma}{\rightarrow} S_b$). For example, in Figure 6 (PT=[G_{R1}, $G_{L4,}$]$\overset{\sigma}{\rightarrow}$Act-1) are selection comparable nodes. Since there is a path between PT=[G_{R1}, $G_{L4,}$] and Act-1.
PT=[G_{R1}, $G_{L4,}$]$\rightarrow$Act-2 are not selection comparable nodes. However PT=[G_{R1}, $G_{L4,}$]$\rightarrow$(Act-2 instances) are selection comparable nodes. The same applies to PT=[G_{R1}, $G_{L4,}$]$\rightarrow$(Act-3).
P Join ($^{P}\times$): The idea behind the extended Join is to combine relations with common high order attributes not only at the top level but also at the subschema level. Let R be the relational relation schema and S be the schema tree of R, the path $P_r = (M_1 ... M_k)$ is a join-path of R if M_1 is a child of root (S) and M_k is a non-leaf node of S. Path expressions describe routes along the composition hierarchy and expressions describe links between attribute domains. The same attribute names in two join relations may appear in multiple sub-trees. The P join can be extended with multiple path joins, which exploit the more general situation. Considering Figure 5 and Figure 6 there is a path between $R_{AND\text{-}Split}$, (14), and $R_{AND\text{-}Join}$ (15), noted as (Act-2. Act-2 instances) which is the joint point of the two relations. The result of the Join operation between is defined by (16) and presented in Figure 7.

7. CONCLUSIONS

An initiatory framework for representing and querying a temporal workflow schema and the evolution of its states over the time has been presented. The innovation in our framework is the separable representation of a temporal workflow schema and its activity states while paths between activities are also encoded and can be queried with the aid of post relational operators.

The framework is currently extended in the direction of developing a complete and sound post relational algebra and query mechanism [9] that will enable the engagement of state and activity relations, showing thus the evolution [10], [11] of objects and their roles over the time while following a fixed but arbitrary path policy.

References

[1] Panagos, E., and Rabinovich M., Reducing escalation- related costs in WFMS, NATO Advanced Study Institue on Workflow Management Systems and Interoperability, Springer Verlag, 1997.

[2] Chountas, P., Petrounias, I., Representation of Definite, Indefinite and Infinite Temporal Information, Proceedings of (IDEAS'2000), *IEEE Computer Society Press-Digital Library*, pp 167-178, 2000.
[3] Workflow Management Coalition, Terminology and Glossary: A Workflow Management Coalition, Brussels, 1996.
[4] Koubarakis, M., *Database Models for Infinite and Indefinite Temporal Information*, Information Systems, Vol. 19, No. 2, pages 141-173, 1994.
[5] Dekhtyar, A., Ross, R.,. Subrahmanian, V. S, *TATA Probabilistic Temporal Databases, I: Algebra*, Technical Report CS-TR-3987, Department of Computer Science, University of Maryland, USA, pp 1-81 1999.
[6] 6. Petrounias, I., *A Conceptual Development Framework for Temporal Information Systems*, Proceedings of Conceptual Modelling - ER '97, Springer Verlag, pp 43-56, 1997.
[7] H. Liu, Ramamohanarao K., *Algebraic Equivalences Among Nested Relational Operations,* Proceedings, of ACM International Conference on Information and Knowledge Management (CIKM), pp 234-243.
[8] M. Kifer, W. Kim, Y. Sagiv, *Querying Object Oriented Databases*, Proceedings of ACM Symposium on Principles of Database Systems (PODS), pp 393-402, 1992.
[9] Chountas, P, Petrounias, I., Representing Temporal and Factual Uncertainty, Proceedings of FQAS 2000, in Advances in Soft Computing, Physica-Verlag Heidelberg, pp 161-170, 2000.
[10] Atanassov, K., Gargov. G., Elements of Intuitionistic fuzzy logic, Fuzzy sets and Systems Vol. 95,1998, No. 1, 39-52.
[11] Atanassov K., Temporal Intuitionistic Fuzzy Relations, Proceedings of FQAS 2000, in Advances in Soft Computing, Physica-Verlag Heidelberg 153-160, 2000.