Conceptual Modeling of Temporal Clinical Workflows*

Carlo Combi
Department of Computer Science
University of Verona – Italy
combi@sci.univr.it

Matteo Gozzi
Department of Computer Science
University of Verona – Italy
gozzi@sci.univr.it

Jose M. Juarez
Department of Information and
Communication Engineering
Universidad de Murcia – Spain
jmjuarez@dif.um.es

Barbara Oliboni
Department of Computer Science
University of Verona – Italy
oliboni@sci.univr.it

Giuseppe Pozzi
Department of Electronics and Information
Politecnico of Milano – Italy
giuseppe.pozzi@polimi.it

Abstract

The diffusion of clinical guidelines to describe the proper way to deal with patients' situations is spreading out and opens new issues in the context of modeling and managing (temporal) information about medical activities. Guidelines can be seen as processes describing the sequence of activities to be executed, and thus approaches proposed in the business context can be used to model them. In this paper, we propose a general conceptual workflow model, considering both activities and their temporal properties, and focus on the representation of clinical guidelines by the proposed model.

1 Introduction

Clinical guidelines describe, usually in natural language, the recommended behavior for a medical team, the activities to apply to a patient, and their fulfillment with respect to the time and to the health state of the patient. Clinical guidelines are narrative documents without a fixed organization. The lack of structure, consistency and the ambiguity of these documents could be managed by computer systems to improve the clinical practice. Due to the fact that the specification of a guideline can be very long, some information can be spread along the document, and the suggested sequence of medical activities can be difficult to understand. To this regard, several attempts have been made to develop computer-based systems, called Computer-Interpretable clinical Guidelines (CIGs), to assist health providers in charting a given patient course through a whole guideline [9].

Finding a sound technique for representing guidelines is an important and difficult issue, requiring to model several elements such as actions, patient's information, decisions to be taken, constraints between activities, and temporal properties. In particular, the need of representing and reasoning with time is crucial when guidelines are used in critical contexts, for instance when physicians have to schedule different drug regimens in specific temporal intervals and several tasks have to be coordinated. Some examples of temporal aspects to be considered are: durations of an activity (e.g., *4 hours for a treatment*), instantaneous events (e.g., *heart attack event*), constraints (e.g., *maximum delay of 5 days between the stroke event and the drug administration*), periodic activities (e.g., *radiation therapy from Monday to Friday for 25 days*), required delays (e.g., *clinical analysis must be re-checked after at least 1 day from the first check*). Moreover, there can also be some implicit temporal constraints that must be considered when a guideline is modeled, such as synchronization of activities (e.g., *clinical assessment must be performed by the physician after completing all the tests*), and serialization of activities (e.g., *haematology checks must be performed after glycerol ad-*

*This work was partially supported by the Spanish MEC under the FPU national plan (grant ref. AP2003-4476)

14th International Symposium on Temporal Representation and Reasoning (TIME'07
0-7695-2836-8/07 $20.00 © 2007 IEEE

ministration).

In the context of temporal reasoning, the literature deeply considered some of the aspects introduced above. In the specific field of clinical guidelines, several research groups proposed languages and tools for modeling temporal information (among the others, we mention here Arden Syntax, Asbru, Dharma, GEM, GLIF, Prestige, PRODIGY, PROforma, EON [23, 24, 26, 28]). All of these languages deal with expressing process semantics, partially including temporal constraints, but generally they do not provide full workflow capabilities, as, for example, notification mechanisms, actor/role support, and resource constraints. On the other hand, unlike from classical workflow-based systems focusing on organizational aspects, they focus on modeling medical domain knowledge [4].

However, as a matter of fact, guidelines are very similar to business processes and have the goal of managing in the correct way the patient's situation, at every instant, for every condition. Clinical guidelines describe flexible and collaborative processes, structuring them in activities (task) which always require human intervention.

Workflow models can be applied to consider all the guideline aspects: tasks, execution flows, participants, conditions, scheduling and temporal properties [6, 7]. Moreover, these models could also consider other temporal issues related to actor's unavailability (e.g. vacations, holidays, and periodic work shifts of physicians and nurses) [8]. For these reasons, we consider workflow models a suitable approach to manage all the guideline aspects, further enriching processes and related information.

According to this scenario, the goal of the paper is twofold:

- we highlight the most important temporal aspects which need to be properly modeled and managed when supporting clinical guidelines;
- we propose a new temporal model for the conceptual design of workflows: it is a general workflow conceptual model allowing the designer to express real world temporal requirements, as those in the medical context of clinical guidelines.

The paper is organized as follows: Section 2 introduces a motivating example we shall use throughout the paper. Section 3 presents a conceptual model, mainly focusing on the representation of activities and flows. Section 4 defines all the temporal issues of the model (e.g., temporal attributes and types of constraints). Section 5 discusses the problem of determining the consistency of temporal workflow schemata. Section 6 describes the related work, compares our proposal with other conceptual models proposed in the literature, and discusses existing work dealing with formalisms capable of specifying the temporal aspects involved in modeling clinical guidelines. Section 7 describes the main features of a prototype we designed and implemented supporting the proposed conceptual modeling. Finally, Section 8 sketches out some concluding remarks.

2 A Motivating Example: Stroke Prevention and Management

Throughout the paper, we consider the following fragment of an Italian guideline for stroke prevention and management (SPREAD) [25], and use it as a motivating example:

> ***Recommendation 10.32:*** *For the treatment of endocranial hypertension, the following actions are recommended: Mannitol should be administered (0.25 to 0.5 g/kg for 4 hours) for patients with elevated endocranial hypertension. Due to the known rebound effect, mannitol has to be used for less than 5 days after the stroke event. Glycerol is usually administered by parenteral route (250 mL of 10% glycerol in 30-60 min every 6 hours). The oral administration (50 mL of 10% solution every 6 hours) is also possible. [...] During therapy with glycerol, haematology should be systematically controlled since it may induce haemolysis. [...] Hypocapnia causes cerebral vasoconstriction, almost immediate decrease of cerebral flow and decreased endocranial pressures after 30 min. At the same time as the other treatments, a decrease of pCO2 to 30-35 mm Hg can be obtained with constant hyperventilation using a volume of 12-14 mL/kg and reduces the endocranial pressured by 25%-30%.*

In a general guideline, temporal information about organizational problems are not specified, but can be useful in order to manage in a concert way the execution of clinical processes. To complete the considered example and show the capabilities of our proposal, we suppose to add the following activities and constraints:

- Before the hyperventilation task, the 3 ventilation machines must be supervised and the operators spend at least 10 minutes for checking the availability. When at least 1 machine is checked, then the next task can be performed.
- After the treatments, endocranial hypertension must be re-checked after at least one day from the first check.
- The re-checking of endocranial hypertension is a non-critical activity and it can be executed between Monday and Saturday of every week.

- The registration of treatment data cannot be performed during Christmas Day.

In the following we will show how our conceptual model can suitably describe temporal aspects highlighted in both the guideline fragment and organization requirements, and we will discuss all the atemporal and temporal constructs of the proposed model. Figure 1 depicts the conceptual schema of the scenario related to the considered portion of the SPREAD guideline[1].

3 Atemporal Aspects of the Workflow Model

A conceptual workflow model describes a process (providing its schema), by activities (also known as tasks), execution flows, and criteria for task assignment to executing agents [5]. Instances of process models are known as cases. As conceptual model of reference throughout the paper we consider the atemporal model proposed in [5], which is not influenced by any particular commercial Workflow Management System (*WfMS*) and is one of the models closest to the recommendations of the Workflow Management Coalition (*WfMC*) [30].

According to our model, a schema is a directed graph, called *Workflow Graph*. Nodes correspond to activities and edges represent control flows that define the execution order of activities.

Definition 1. ***(Workflow Graph)*** *A Workflow Graph $WG = (N, T, C, C_S, C_E, F)$ is a six-uple such as:*

- *N is a finite set of nodes*
- *$T \subseteq N$ is a finite set of tasks*
- *$C \subseteq N$ is a finite set of connectors*
- *$C_S \subseteq C$ and $|C_S| = 1$ is the Startcase set*
- *$C_E \subseteq C$ is the Endcase set*
- *$N = T \cup C$ and*
- *$F \subset N \times N$ is the control flow relation*

The relation F *defines a directed graph with nodes* N *and edges* F.

N is the set of all the activities that belong to the process. There are two different activity types: *task* and *connector*. Tasks are the elementary work units that collectively achieve the process goal. They are either automated or manual activities that are performed by assigned agents.

[1] For sake of simplicity we consider here only a portion of the guideline. The proposed model suffices to represent the entire guideline.

A connector can be initial (*startcase*), final (*endcase*) or intermediate. Each workflow graph must have only one *startcase* and at least one *endcase*. When any *endcase* is executed, the execution of the process is completed: more precisely, when the earliest *endcase* is performed, the execution is considered successful and all the other tasks, which are still active, are cancelled. Graphically, *startcase* and *endcase* are represented in the model by two horizontal parallel lines (see Figure 1).

Intermediate connectors are exploited to model different execution paths (parallel, alternative, conditional). Connector nodes are the activities executed directly by the *WfMS* to achieve a correct and ordered sequence execution for tasks. Therefore, the graphical representation (a circle) of intermediate connectors in a workflow graph is different from task representation (a box) to underline the conceptual dissimilarity of corresponding activities. The flow relation *F* defines the connection between two given nodes. In the following, we will present the syntactic and semantic properties of all the workflow graph components.

Task: it is the basic modeling unit in a schema and represents the atomic unit of work. A task has one incoming edge and one outgoing edge and is graphically represented as a box.

Multitask: it stands for a set of tasks performing exactly the same job in parallel that are assigned to different agents (e.g., physicians and nurses). Each multitask is associated to a value k standing for the number of task instances that start when the multitask is started. The value n ($k > n$) represents how many instances must complete their execution to consider the multitask completed. Subsequent ending of components has no effect. Figure 2(a) shows an example of multitask: in this case three (3) instances of the activity `T6` *Check ventilation machine availability* are launched, one for each ventilation machine. When at least one machine is checked, then the task is considered completed and the other instances are automatically aborted by the *WfMS*. The multitask shown in Figure 2 also has a temporal property that will be discussed in the next section.

As in the *WfMC* Reference Model [30], our model has two connector types: *split* and *join*.

Split connectors are routing connectors specifying that, after the completion of the predecessor activity, several successor activities have to be considered for the execution. The set of activities that start their execution is given by the features of each *split* connector. A *split* connector can be:

Total: it specifies the independent execution of concurrent paths within a workflow graph. After a total connector,

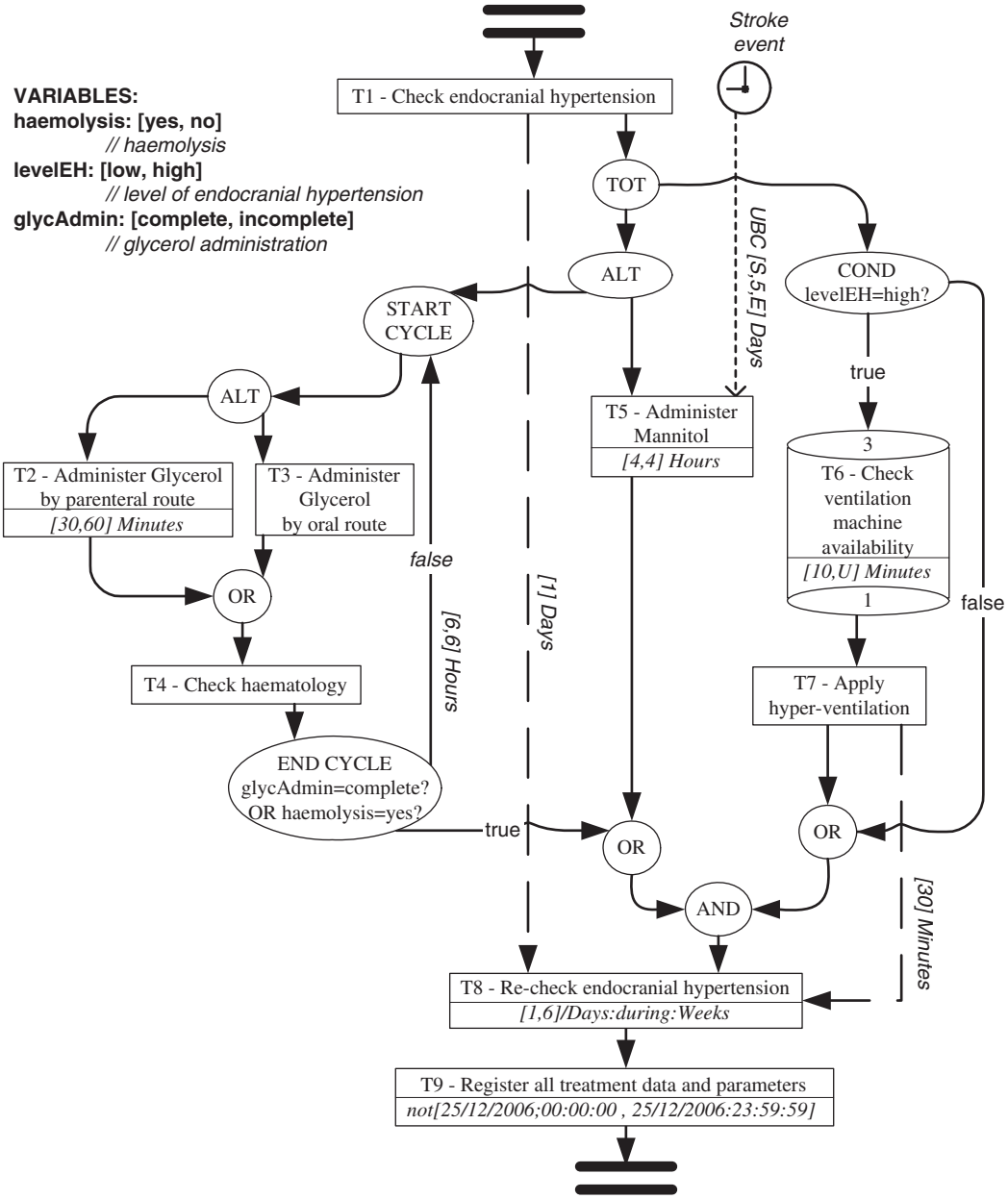

Figure 1. The workflow schema of a fragment of the SPREAD **guideline, for the treatment of endocranial hypertension.**

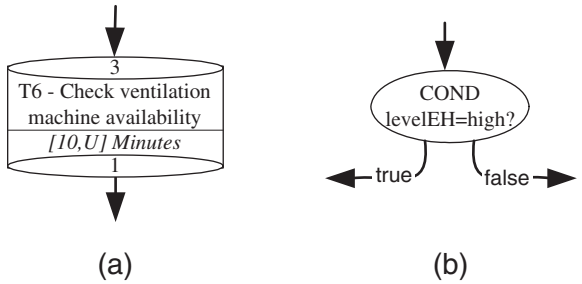

Figure 2. Representation of a multitask (a) and a conditional connector (b)

all successors of its are ready for execution (see, for example, the node TOT in Figure 1).

Alternative: it stands for a choice node. It has two or more outgoing execution flows resulting in mutually exclusive alternative paths. This ensures that only one alternative successor node is selected at run-time. For example, in Figure 1, tasks T2 and T3 belong to alternative paths, and thus are preceded by the alternative node ALT.

Conditional: it represents alternative paths in the schema, depending on the state of some workflow variables. Any conditional successor is associated to the value of a boolean expression called *condition*. Figure 2 provides an example of a conditional connector in which the evaluation of the condition *"levelEH=high?"* determines if the following task to be executed is associated to the corresponding true or false values.

Endcycle: it has the same properties as the conditional connector. Each cycle in this model is bounded between a *startcycle* and an *endcycle* connectors. Only these two components allow the design of a cycle inside the graph. Additionally, the *endcycle* defines the suitable cycle attributes: *iterations* and *deadline*. Iterations are represented as an interval expressed as the minimum and maximum number of allowed iterations. Deadline is the maximum duration allowed for completing the cycle execution. At least one of these two attributes must be set by the workflow designer to ensure the cycle termination.

Join connectors are nodes with two or more incoming edges and only one outgoing edge. Indeed, *join* connectors merge more execution flows into one single flow. There are three different kinds of flows join:

And: it joins at least two or more concurrent execution paths. It plays the role of a synchronizer, as it becomes ready only after the completion of all predecessors of its.

Or: it is applied to join two or more alternative paths into one path. It becomes ready after the completion of the earliest predecessor.

Startcycle: it shows the starting point of a cycle. The *startcycle* always has two incoming edges: one connects the predecessor to the *startcycle*, while the other one connects the corresponding *endcycle* to the *startcycle*.

Figure 1 shows a cycle as a subgraph of the schema with START CYCLE as starting node and END CYCLE as ending node of the subgraph. All the activities belonging to the paths from START CYCLE to END CYCLE are repeated until the condition *"glycAdmin=complete OR haemolysis=yes"* assumes the value TRUE. Inside the cycle, the ALT connector is the beginning of two alternative paths, and the OR connector joins them.

4 Modeling Temporal Aspects

Time aspects, as highlighted even in the provided guideline fragment, are very relevant in the definition of a workflow schema and, thus, *WfMS*s must be able to manage the needed information about a process, its time restrictions, and its actual time requirements from process designers and managers [12]. This information must be represented within the process definition.

In this paper, we describe how time constraints (e.g., upper and lower bound of tasks durations, minimum and maximum delays between activity executions, fixed-date and periodic constraints) can be represented during the conceptual process definition. Therefore, the basic temporal concepts used in a process are introduced and the conceptual modeling of such kind of temporal information is presented.

Our model considers instants and durations as elementary temporal types. Instants are points on the time domain, while durations are lengths on the time domain. Intervals are derived types and can be defined either as a duration after an instant or as a duration before an instant or as the distance between starting and ending instants [6, 7]. Instants are represented through timestamps: each timestamp is defined with a granularity. A time granularity intuitively partitions the time domain in a sequence of granules, e.g., sets of points on the time domain. For example, each granule of the granularity day specifies the set of time points included in a particular day.

We adopt the time granularity concepts formalized in [1], where the authors introduce a mathematical characterization of time granularities. The proposed formalism covers standard granularities like Days, Weeks and Years, bounded granularities like Years-since-2000, and

specialized granularities like `Business-days` and `Business-months`.

4.1 Events

An *event* is something that may occur during the execution of a process and whose semantic is not negligible. Events represent occurrences of facts which are not modeled directly within the workflow schema and which happen at some time points. These occurrences are linked to the temporal behavior of workflow activities. To do this, we explicitly represent the concept of *event*, similarly to the approach taken in the Business Process Modeling Notation [22].

In our model, an *event* is represented as a rounded clock and indicates a time instant that could bias the execution of activities, as depicted in Figure 1. Indeed, *event*s could be linked to activities by means of *relative* constraints or *required delay*s, as detailed in the following.

4.2 Durations and Delays

The duration is a temporal property of both nodes and edges. A node duration represents the temporal span of the corresponding activity and is the temporal distance between the starting and ending instants of the activity. The edge duration, called *delay*, denotes the interval between the ending instant of the predecessor and the starting instant of the successor, e.g., the duration at disposal of the *WfMS* to arrange the start of an activity after the end of the previous one.

In the real world, workflow designers cannot use precise values for durations of nodes and edges. For example, tasks can be performed by human agents, and generally the duration of a task cannot be precisely known at schema design time; on the other hand, even for internal activities (e.g., a split) it is not possible to precisely know how much the *WfMS* will take to perform them, depending on the work load of the system. Therefore, allowed durations should be expressed in a conceptual model by using intervals: our conceptual model describes all the durations by an attribute having the form `[MinDuration,MaxDuration] Granule`. Range bounds represent the minimum and maximum allowed durations.

In some cases, only maximum or minimum duration can be defined: consequently intervals could be only partially defined. In our model, duration and delay bounds are integers if the interval is completely specified, otherwise we use the symbol U to represent undefined values. For instance, the multitask shown in Figure 2 has an undefined maximum duration.

In our conceptual process model, duration attributes are mandatory for each component. If the workflow designer does not set the duration of a component, the default value for duration is `[0,0]MinGranularity`, where `MinGranularity` is the finest granularity of that schema. In this case, the activity is considered as instantaneous.

4.3 Temporal Constraints

Beside basic temporal constraints, expressed through durations, and required delays, our conceptual model allows one to express several kinds of temporal constraints: from the business perspective they are defined by laws and regulations, business policies, common practices, as well as mutual agreements and expectations related to efficiency/productivity of the business practice [20]. In general, temporal constraints are complex enough to be captured separately as an aspect of process modeling rather than being covered by the workflow execution properties [13, 14]. Differently from the duration attributes, temporal constraints are not mandatory for each workflow component: they model additional temporal properties and must be controlled by the *WfMS*.

We classify temporal constraints for process modeling as *relative* constraints, *absolute* constraints and *periodic* constraints, as in the following.

Relative Constraints

A *relative* constraint limits the time distance (duration) between the starting/ending instants of two non-subsequent tasks. Our model provides two types of *relative* constraints, expressed as:

Upper Bound Constraint
`UBC[`I_F`,TimeDistance,`I_L`]Granule` denotes that *TimeDistance* must be the maximum range between the starting or ending execution instant (I_F) of the first task and the starting or ending execution instant (I_L) of the last task.

Lower Bound Constraint
`LBC[`I_F`,TimeDistance,`I_L`]Granule` denotes that *TimeDistance* must be the minimum range between the starting or ending execution instant (I_F) of the first task and the starting or ending execution instant (I_L) of the last task.

An *upper bound* constraint can model a deadline, that corresponds to the maximum allowable execution time for activities. Indeed, if the first activity of the *upper bound* constraint is the *startcase* of the schema, then the *TimeDistance* value represents the deadline of the last activity of the *upper bound* constraint.

Figure 1 provides an example of a *relative* constraint, which fixes to `5 days` the maximum time distance between the stroke event and the end of mannitol administration, as required by the considered fragment of the guideline.

Absolute Constraints

An *absolute* constraint represents a time interval during which an activity can be performed. Interval bounds are expressed by timestamps: bounds of an *absolute* constraint are independent from the case execution starting timestamp. The span of the time interval for the *absolute* constraint must be greater than or equal to the maximum duration defined for the corresponding activity.

An *absolute* constraint can also be used to represent a time interval during which an activity cannot be executed: in this case, the key "*not*" must be specified before the specification of the time interval.

Figure 1 provides an example of an *absolute* constraint which specifies that the task `T9` cannot be executed during Christmas Day.

Periodic Constraints

A periodic constraint represents a periodic time interval during which an activity can be started. We represent periodic intervals by the notation proposed by Leban at el. in [18]: this notation formally defines sets of time intervals and provides a sound description of periodic behaviors.

Figure 1 provides an example of a *periodic* constraint which specifies that the task `T8` can be executed between Monday and Saturday of every week.

4.4 Required Delays

Required delays represent the minimal time distance between two non-subsequent activities: this is not a temporal constraint because it cannot be violated as it imposes the *WfMS* to wait until the *required delay* expired.

Suppose that a *required delay* forces at least 1 hour between the non-subsequent tasks A and B. If B could start 1 hour after the end of A, then B starts immediately. Otherwise, if B could start less than 1 hour after the end of A, then B must wait until 1 hour expired after the end of A. *Required delay*s of subsequent activities have the same meaning as minimal delays expressed in the edge that connects the two activities. *Required delay*s are depicted in our model as large dashed edges that connect two non-subsequent nodes. Edges are labeled by the time distance between the two activities and the corresponding granule `[TimeDistance] Granule`.

Figure 1 provides two examples of *required delays*. The first one imposes that endocranial hypertension has to be re-checked at least one day after the previous check. The second one is derived by the guideline specification that the effect of hyperventilation is assured after at least 30 min and then the endocranial hypertension must be re-checked after this span.

*Required delay*s are conceptually different from *relative* constraints: indeed, these latter represent bounds that need an exception handling mechanism if they are violated. Instead, a *required delay* represents a time span the *WfMS* has to wait for, until the *required delay* expires. Let us reconsider the required delay depicted in Figure 1 representing that the endocranial hypertension has to be re-checked (task `T8`) at least one day after the previous check (task `T1`). In this case, if all the tasks, including those related to the *WfMS*, preceding the task `T8` have been done in less than 24 hours, then the *WfMS* has to wait for some span of time (to reach a delay of 24 hours from `T1`), before proceeding with task `T8`. Changing this *required delay* into an apparently equivalent *relative* constraint `LBC[`E`,1,`S`]Day`, would mean that, if all the tasks, including those related to the *WfMS*, preceding the task `T8` have been done in less than 24 hours, then the relative constraint would be violated and the *WfMS* should manage a workflow exception, as the given case cannot finish according to the defined temporal constraints.

5 Verifying the Consistency of a Workflow Schema

The successful completion of a process often depends on the correctness of the temporal aspects modeled at design time. If temporal constraints cannot be satisfied, the process cannot be successfully performed. Temporal evaluations and simulations are needed to state whether the specified temporal constraints can be satisfied by some workflow instances, i.e. the workflow schema is somehow consistent.

As each workflow schema may contain several alternative or conditional split connectors, we could have several possible execution paths for the same workflow schema: consistency must be verified for each of these possible paths. The consistency of each of these workflow paths can be described as a simple temporal problem (STP), which is known in literature to be a tractable problem when only one granularity is used for expressing temporal constraints [10]. As for the general case of single-interval constraints given at different granularities, in [2], Bettini et al. show that checking temporal consistency is an NP-hard problem.

A simplifying approach we could take when verifying temporal constraints at different granularities is that of considering only granularities without gaps inside and applying an *unanchored interpretation* to temporal constraints at different granularities: it means, for example, that the granularity `Business-months` would not be allowed, as tem-

poral constraints at this granularity correspond to disjunction of intervals, when expressed on the basic timeline. The *unanchored interpretation* means that, for example, a temporal constraint as LBC[E,1,S]Day corresponds to the constraint LBC[E,24,S]Hour. Despite a reduction in the expressivity of the model (we could constrain tasks to be executed within 24 hours, but not in the same day, for example), this way each execution path could be expressed as a STP and its consistency verified in polynomial time.

It is worth noting that, as we are dealing with a conceptual model, the consistency of the temporal workflow schema is checked at the design time and is in some sense "static", as opposed to the dynamic consistency checking performed during a workflow execution [3]. Moreover, according to this approach, the consistency of a workflow schema at the conceptual level could be interpreted from different points of view: indeed, the consistency of the overall workflow schema could be derived from the consistency of each workflow path, considered separately. However, the consistency of each workflow path could be not enough at the conceptual level: a more restrictive view could require, for example, that any duration of a given task allowed by a given temporal constraint is consistent with each workflow path (or with most paths of the schema).

6 Related Work

In this section we compare our model with both main temporal process models and main Computer-interpretable clinical guidelines (CIGs).

6.1 Temporal Process Models

Several process models have been developed based on different modeling concepts (e.g. Petri net variants, precedence graph models, precedence graphs with control nodes, state charts, control structure based models, to mention few of them) and on different representation models (programming language style, text based models, simple graphical flow models, structured graphs) [5, 15, 16, 17, 19, 29].

With respect to the different points of view, each of the listed modeling and representation concepts has its own characteristics, strengths and weaknesses. In the following we compare our conceptual model with other similar proposals from the literature. Although these models have different features, they follow the main directives from the *WfMC*, as in [30] and [31].

All of these models describe the processes using directed acyclic graph (*DAG*), in which nodes represent tasks and oriented edges represent execution flows. They provide the capability of specifying parallel execution flows and alternative execution flows, as recommended in [30]. *DAG*s are too strict and inflexible to represent complex processes and iterative executions, such as cycles, cannot be mapped by *DAG*s.

Eder et al. in [15] specify the *Timed Workflow Graph (*TWG*)* to represent the temporal properties of tasks named "activity nodes". *TWG* provides a notation indicating different kinds of execution flows, such as parallel or alternative ones, but connector nodes are not considered. Without connectors this model cannot define the temporal behavior of *WfMS* activities, such as execution delays. Each task of *TWG* is described by a single value for duration. Temporal ranges, such as minimum and maximum durations, are not allowed and thus the representation of some real-world temporal aspects of processes are limited. *TWG* can provide one temporal constraint only, called deadline, and represents an upper bound for the execution of the task. Finally, *TWG* does not provide multiple granularity management: all the temporal attributes are expressed by absolute integers representing a granule temporal units.

Marjanovic et al. define in [20] a conceptual model that classifies temporal aspects of schemata as: *basic temporal constraint*, *limited duration constraint*, *deadline constraint* and *interdependent temporal constraint*. A *basic temporal constraint* limits the expected duration of one single task. The duration is specified by using a closed interval. No undefined bounds are admitted. A *limited duration constraint* is an upper bound for the duration of the execution: our proposal does not allow one to directly specify this kind of limitation, but it can be expressed by *relative* constraints between the *startcase* node and each *endcase* node of the workflow graph: indeed, *relative* constraints represent (the minimum and) the maximum temporal distance between two non-subsequent tasks. A *deadline constraint*, in terms of absolute time, limits when a task can start or can be completed during an execution of a case: this constraint corresponds to the absolute constraint of our model. An *interdependent temporal constraint* limits the time distance between two tasks in a schema: it has the same meaning of the relative constraint in our model. The model by Marjanovic et al. does not permit any delay interval between subsequent activities: the completion instant of a task corresponds to the starting instant of the subsequent task.

In the model of Bettini et al. [3], the nodes of a workflow graph are not tasks, but correspond to temporal instants. Every task is represented by two nodes: the starting instant and the ending instant. Every edge in the workflow graph represents the temporal distance between two nodes: the label of the edge is the minimum and maximum time distance between the connected nodes. If the nodes correspond to the starting and ending instants of a task, the interval of the connecting edge is the duration of the execution of the task. In the same way, if the nodes correspond to the ending instant of a task and the starting instant of another task, the connecting edge represents the delay be-

tween the executions of the two tasks. With this formalism, the model makes no difference between edges representing execution flows and edges limiting the time distance between nodes, as this proposal focuses on the definition of schedules (i.e., sets of suitable task executions) defined according to the temporal constraints specified for the given workflow. This model gives the possibility of defining subsequent nodes without imposing which one should be the first. To model this aspect, Bettini et al. introduce negative integers as lower bound of distance intervals. In this way, the nodes are linked by a temporal distance constraint, but the execution order of the tasks is not imposed. In our model the time interval cannot be defined by negative integers, but two alternative paths can be composed to map the unfixed order of execution. Finally, the model by Bettini et al. manages granularities for time intervals, according to the model for granularities proposed in [1].

6.2 Computer-Interpretable Guideline Models

Computer-interpretable clinical guidelines (*CIG*s) are being developed to support decision-making during clinical encounters [9]. *CIG*s represent guideline by "Task-Network Models", but they differ in their approaches for various modeling concepts. We concentrate on the proposals that perform the most complex temporal abstractions.

Asbru [11] represents clinical guidelines as time-oriented skeletal-plans. A skeletal plan captures the essence of a procedure, leaving execution-time flexibility in the achievement of particular intentions (i.e., goals). Asbru enriches skeletal plans (1) by characterizing plan attributes such as intentions, conditions, and affects, (2) by adding a set of plans (sequential, parallel, any-order, unordered, subplans), and (3) by defining the temporal dimension of states and plans. Uncertainty in both temporal scopes and parameters can be expressed by the begin, the end and the duration of the intervals. Furthermore, no time point for the begin and the end is defined, but shifts from an arbitrarily defined reference point. This point can be individually assigned for any interval. In this model there is no other temporal constraint to enrich the regulation of plan executions. For instance, there is no interdependent temporal constraint to limit the time distance between two non-subsequent plans, as the *relative* constraint does in our model.

EON [21] uses a task-based approach to define decision-support services and deals with the guideline model as the core of an extensible set of models (temporal model, concept model, patient data model, expression language, guideline model). The temporal model depicts a class hierarchy with different kinds of time stamps (e.g., time point, duration and time interval). The model also includes absolute time points with fuzzy borders, relative time points (e.g., today), fuzzy durations, and time intervals. As for the Asbru model, EON provides no temporal constraints between non-subsequent tasks.

GLARE [27] represents clinical guidelines using a language that aims at balancing expressiveness and complexity. GLARE distinguishes between *atomic* and composite actions (*plans*): atomic actions represent simple steps in a guideline; plans represent actions which can be defined in terms of their components via the *has-part* relation. A *control relation* establishes which action can be executed next, and in which order (sequence, controlled, alternative and repetition). The formalism of GLARE allows one to represent the (minimum and maximum) duration of each atomic action. Temporal constraints can also be associated to control relations between actions. In fact, in a controlled relation, one can specify the minimum and/or maximum distance between any pair of endpoints of the involved actions. However, there is no conceptual distinction between a time distance that bounds two endpoints (as *relative* constraints of our model) and a minimum elapsed amount of time between two non-subsequent endpoints, as the *required delay* does in our model. Finally, GLARE does not include any external *event* or any similar entity to formalizes something external to the process and that is linked to the temporal behavior of actions.

7 A Prototype Implementation

In this section, we focus on the description of our prototype named *Temporal Workflow Analyzer* (*TWA*), developed to support workflow modeling and time management at workflow design time. The main reason for developing *TWA* is the creation of a tool supporting the design and analysis of temporal workflow schemata. Due to the compatibility of our conceptual model with the main directives suggested by the *WfMC* [30], this prototype could be useful for all the applications that require workflow schemata close to the *WfMC* directives. Indeed, our prototype generates *XML* specifications that are composed according to an *XML*-Schema that extends the *XML*-Schema proposed by the *WfMC*: thus, external workflow applications capable of managing the *WfMC*'s *XML* descriptions may also employ the *XML* files generated by our prototype.

Another reason for developing *TWA* is to show, as a proof-of-concept, that our theoretical and methodological proposal can be applied in a tool supporting the specification of real world clinical workflow schemata. Figure 3 shows the main window of *TWA* and reports a workflow schema specifying how to deal with the admission of a new patient into a hospital ward: it is composed by practical activities related to searching for/preparing a clean room and by administration activities dealing with the preparation of the clinical documentation.

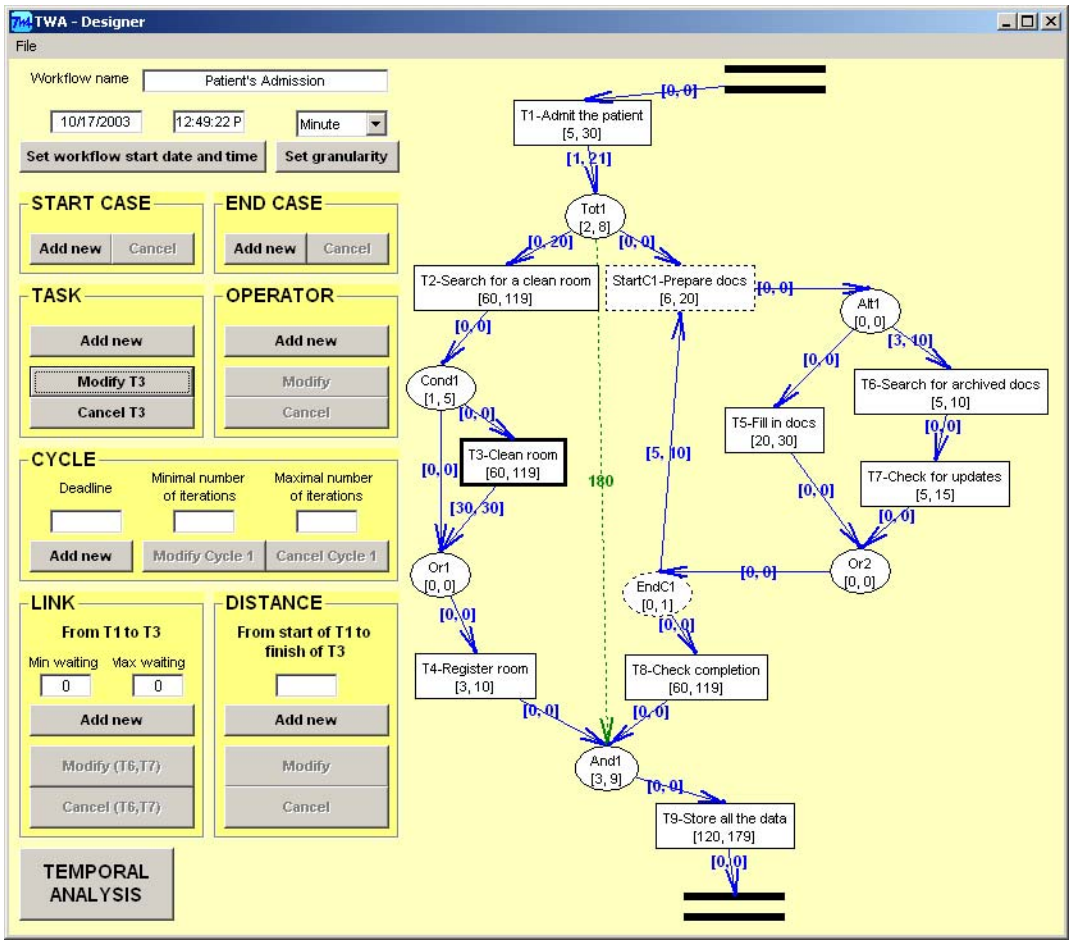

Figure 3. The main window of *TWA-Designer*.

7.1 *TWA* Functionalities

The *TWA* tool provides functionalities that allow the user to build conceptual schemata and to analyze the consistency of temporal constraints, according to the simplified (and less expressive) approach mentioned in Section 5. These functionalities include:

- **Workflow specification**:
  - *Creation:* building up a new workflow schema that includes the creation of tasks, connectors and execution flows. All the temporal constraints supplied by our conceptual model can be defined.
  - *Modification:* insertion, deletion and modification of workflow schemata.
  - *Deletion:* deletion of a workflow schema.
- **Workflow correctness verification**: the program assists the workflow designer in instantaneously identifying any syntactical error.
- **Temporal analysis**: The workflow schema is translated into a temporal network and the workflow consistency is evaluated (path by path).
- ***XML* representation of a workflow schema**: The workflow schema specification is translated into an *XML* representation extending the *XML*-Schema proposed by the *WfMC*.

The prototype includes the modules named *TWA-designer* and *TWA-simulator*, described below.

7.2 *TWA*-Designer

The *TWA-designer* is a tool to design temporal workflow schemata. The main window (Figure 3) can be divided into two parts: on the left side, the forms allow the user to insert, modify and delete all the constructs, and to define all the temporal properties; on the right side, the schema of a

workflow process is shown. All the components are sensitive to mouse roll over and information, such as variable values or cycles properties, can be displayed.

During schema implementation, the application tests at run time the syntactic correctness of the schema: the user is supported to find out design errors. If the schema is correct, the button "Temporal Analysis" is enabled and the following prototype module can be accessed.

7.3 *TWA*-Simulator

The *TWA-simulator* translates the workflow schema into temporal constraint networks and performs the temporal analysis: the tool notifies the user about inconsistent situations.

Another important functionality of the *TWA-simulator* is the export of the workflow representation to an *XML* document that is valid and well-formed with respect to the defined *XML*-Schema.

8 Conclusions

In this paper, we proposed a conceptual workflow model capable to represent activities composing a (business) process and the temporal aspects related to them and to their execution. The proposed model collects both classical features of *WfMS*s, and new temporal concepts and properties to manage time and temporal constraints for the considered process. Some of the strengths of the model are the level of abstraction the proposed formalism provides and the suitable integration of temporal aspects with the classical workflow constructs. As for expressiveness, we introduce some new concepts: most of the approaches proposed in the literature and discussed in the previous section formalizes neither external *event*s nor any construct with a semantics similar to that of the proposed *required delay*.

Among the complex application domains, in this paper we focused on the clinical field, where the computer-supported management of clinical guidelines is currently an important and widely investigated research topic. More particularly, we showed the advanced temporal features of our model by representing a fragment of the SPREAD guideline.

As future work, we plan to study in detail the different meanings of temporal consistency for conceptual workflow schemata and focus on supporting the designer with efficient algorithms, allowing one to verify the temporal consistency of the workflow schema at design time.

References

[1] C. Bettini, X. S. Wang, and S. Jajodia. A general framework for time granularity and its application to temporal reasoning. *Ann. Math. Artif. Intell.*, 22(1-2):29–58, 1998.

[2] C. Bettini, X. S. Wang, and S. Jajodia. Solving multi-granularity temporal constraint networks. *Artif. Intell.*, 140(1/2):107–152, 2002.

[3] C. Bettini, X. S. Wang, and S. Jajodia. Temporal reasoning in workflow systems. *Distributed and Parallel Databases*, 11(3):269–306, 2002.

[4] E. D. Browne, M. Schre, and J. R. Warren. Goal-focused self-modifying workflow in the healthcare domain. *Proceedings of the 37th Hawaii International Conference on System Sciences*, 2003.

[5] F. Casati, S. Ceri, B. Pernici, and G. Pozzi. Conceptual modelling of workflows. In M. P. Papazoglou, editor, *OOER*, volume 1021 of *Lecture Notes in Computer Science*, pages 341–354. Springer, 1995.

[6] C. Combi and G. Pozzi. Temporal conceptual modelling of workflows. In I.-Y. Song, S. W. Liddle, T. W. Ling, and P. Scheuermann, editors, *ER*, volume 2813 of *Lecture Notes in Computer Science*, pages 59–76. Springer, 2003.

[7] C. Combi and G. Pozzi. Architectures for a temporal workflow management system. In H. Haddad, A. Omicini, R. L. Wainwright, and L. M. Liebrock, editors, *Proceedings of the 2004 ACM Symposium on Applied Computing (SAC)*, pages 659–666. ACM, 2004.

[8] C. Combi and G. Pozzi. Task scheduling for a temporal workflow management system. In *13th International Symposium on Temporal Representation and Reasoning (TIME 2006)*, pages 61–68. IEEE Computer Society, 2006.

[9] P. A. d. Clercq, J. A. Blom, H. H. Korsten, and A. Hasman. Approaches for creating computer-interpretable guidelines that facilitate decision support. *Artificial Intelligence in Medicine*, 31, 2004.

[10] R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *Artif. Intell.*, 49(1-3):61–95, 1991.

[11] G. Duftschmid, S. Miksch, and W. Gall. Verification of temporal scheduling constraints in clinical practice guidelines. *Artificial Intelligence in Medicine*, 25(2):93–121, 2002.

[12] J. Eder, W. Gruber, and E. Panagos. Temporal modeling of workflows with conditional execution paths. In M. T. Ibrahim, J. Küng, and N. Revell, editors, *DEXA*, volume 1873 of *Lecture Notes in Computer Science*, pages 243–253. Springer, 2000.

[13] J. Eder, E. Panagos, and M. Rabinovich. Time constraints in workflow systems. In M. Jarke and A. Oberweis, editors, *CAiSE*, volume 1626 of *Lecture Notes in Computer Science*, pages 286–300. Springer, 1999.

[14] A. Geppert, D. Tombros, and K. R. Dittrich. Defining the semantics of reactive components in event-driven workflow execution with event histories. *Inf. Syst.*, 23(3-4):235–252, 1998.

[15] Johann Eder and Euthimios Panagos. Managing Time in Workflow Systems. In *Workflow Handbook 2001*, pages 109–132. Workflow Management Coalition (WfMC), 2000.

[16] M. Kamath and K. Ramamritham. Correctness issues in workflow management. *Distributed Systems Engineering*, 3(4):213–221, 1996.

[17] D. Kuo, M. Lawley, C. Liu, and M. E. Orlowska. A model for transactional workflows. In *Australasian Database Conference*, pages 139–146, 1996.

0-7695-2936-8/07 $20.00 © 2007 IEEE

[18] B. Leban, D. McDonald, and D. Forster. A representation for collections of temporal intervals. In *AAAI86*, pages 367–371, 1986.

[19] P. J. Mangan and S. W. Sadiq. A constraint specification aproach to building flexible workflows. *Journal of Research and Practice in Information Technology*, 35(1):21–39, 2003.

[20] O. Marjanovic and M. E. Orlowska. On modeling and verification of temporal constraints in production workflows. *Knowl. Inf. Syst.*, 1(2):157–192, 1999.

[21] M. A. Musen, S. W. Tu, A. K. Das, and Y. Shahar. Eon: A component-based approach to automation of protocol-directed therapy. *Journal of the American Medical Information Association (JAMIA)*, 3(6), 1996.

[22] Object Management Group. Business Process Modeling Notation Specification 1.0, OMG Final Adopted Specification. Technical report, Object Management Group, http://www.bpmn.org, 2006.

[23] S. Panzarasa and M. Stefanelli. Workflow management systems for guideline implementation. *Neurological Sciences*, 27, 2006.

[24] M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, R. A. Greenes, R. Hall, P. D. Johnson, N. Jones, A. Kumar, S. Miksch, S. Quaglini, A. Seyfang, E. H. Shortliffe, and M. Stefanelli. Comparing computer-interpretable guideline models: A case-study approach. *Journal of the American Medical Informatics Association : JAMIA.*, 10(1), 2003.

[25] T. S. Prevention and E. A. D. S. Collaboration. The italian guidelines for stroke prevention. *Neurological Sciences*, 21, 2000.

[26] S. Quaglini and P. Ciccarese. Models for guideline representation. *Neurological Sciences*, 27, 2006.

[27] P. Terenziani, C. Carlini, and S. Montani. Towards a comprehensive treatment of temporal constraints in clinical guidelines. In *TIME*, pages 20–27, 2002.

[28] D. Wang, M. Peleg, S. W. Tu, A. A. Boxwala, R. A. Greenes, V. L. Patel, and E. H. Shortliffe. Representation primitives, process models and patient data in computer-interpretable clinical practice guidelines: A literature review of guideline representation model. *International Journal of Medical Informatics*, 68, 2002.

[29] G. Wirtz, M. Weske, and H. Giese. The ocon approach to workflow modeling in object-oriented systems. *Information Systems Frontiers*, 3(3):357–376, 2001.

[30] Workflow Management Coalition. The Workflow Reference Model. Technical report, Workflow Management Coalition (WfMC), http://www.wfmc.org, 1995.

[31] Workflow Management Coalition. Interface 1 - Process Definition Interchange Process Model. Technical report, Workflow Management Coalition (WfMC), http://www.wfmc.org, 1999.

0-7695-2936-8/07 $20.00