Temporal Aspects in Business Processes – An application to E-Government

Andreas Gehlert, Robert Braun, Werner Esswein
Dresden University of Technology,
Professorship of Information Science, especially System Development
Mommsenstr. 13
01062 Dresden
{gehlert | braun | esswein}@wise.wiwi.tu-dresden.de

Abstract: This paper analyses which temporal aspects need to be recognised in business process modelling. The periodicity and the temporal reference of information objects are discussed in detail. These theoretical considerations are illustrated with examples of the E-Government domain. Furthermore, the findings are implemented into the OMG’s Unified Modelling standard to show the significance of the results for the current modelling practice. Because of the systematic analysis our findings can be transferred to other modelling languages and other problem domains.

1 Introduction

Implementing E-Government solutions such as portals is the last step in a series of phases within the information systems development cycle. In the first phase of this cycle, the problem domain must be analysed to eliminate conceptual problems. To represent the results of this phase, modelling languages must be used that are suited for the specific problem domain. Not specific to but very important for the E-Government domain are various temporal aspects.

Modelling of temporal aspects was addressed during the 70s and the 80s of the last century because of the extensive introduction and use of automated systems (see [RBL88]; [Sc94], p. 284). Research of temporal aspects was concentrated mainly in the fields of temporal databases and real time systems2 (see [Ma91]; [TL91]; [TLW91]; [MH97]; [KJS01]; [KW02]; [Ko03]). Consequently, temporal aspects in these areas are nowadays theoretically well understood.

---

1 The insights described in this paper resulted from a project conducted with the Saxon Ministry of Finance (http://wise.wiwi.tu-dresden.de/systementwicklung/projekte/nfs/index.html).
2 The term real time system is used in its broadest sense including event-driven and time-driven systems (see [Do99], p. 58).
This research led to new and enhanced modelling techniques to account for the specific temporal requirements in the above mentioned areas. These adapted modelling techniques allow the description of the system’s structure on a conceptual level.

Techniques introduced for temporal databases (see [Ko03]) include the Spatio-Temporal Entity-Relationship Model (STER; see [Ko03], pp. 82) and the Temporal Unified Modelling Language (TUML) (see [ST99]). However, all introduced and adapted techniques focus on the system’s structure only. Especially in the field of the Unified Modelling Language (UML) temporal aspects were not introduced in other diagram forms.\(^3\)

The same can be said for modelling real time systems. Again, formal and semiformal languages were extended with temporal modelling issues. Research in this area includes statecharts (see [HP85]; [Ha87]) in their broadest sense and the UML profile Schedulability, Performance and Time Specification (see [OM03b]; for other examples see [Do99], pp. 721).

According to BIDER and JOHANESSEN at least half of the software development industry is involved in business process analysis and design, each of which are the key activities in the software development process (see [BJ02], p. 7). Thus, acknowledging temporal aspects in this field is of increasing importance.

The knowledge acquired in temporal database and real-time modelling is not easily transferable to the field of business process modelling. The main reason for the incompatibility of the temporal concepts used in these two fields are the different modelling domains. Modelling real time systems and temporal databases is a design activity. The result of this phase in the system development process is the specification of a formal (software) system ([So92], p. 173). System analysts with expert knowledge in modelling and system design develop this formal specification.

In contrast, business process modelling is primarily a conceptual activity.\(^4\) The main result of this phase is a semiformal description of the relevant real-world portion. To acquire the specific knowledge about the problem domain modelling experts must cooperate with domain experts. BECKER ET AL concluded that a modelling language suited for this phase must have at least two properties ([BRU00], p. 30). First, the language

\(^3\) This situation changed only slightly with the introduction of the forthcoming UML 2.0 standard. With respect to temporal modelling the timing diagram has been added. It is an “...interaction diagram that shows the change in state or condition of a lifeline (representing a Classifier Instance or Classifier Role) over linear time.” (see [OM03c], p. 16) Furthermore, the new time event concept has been introduced. A time event is an “...event that denotes the time elapsed since the current state was entered.” (see [OM03c], p. 16) While the timing diagram is used to describe the state changes of objects over the time, the time event is modelled in activity diagrams to generate an event based upon a certain time frame (e.g. each month; see [OM03c], pp. 218; especially figure 159).

\(^4\) We use the term conceptual modelling in the sense of WEBER: “Conceptual modelling is an activity undertaken during information systems development and maintenance work to build a representation of selected semantics about some real-world domain. The conceptual model that results is used as a basis for the design and programming tasks that must be undertaken to build or modify an information system.” ([We03], p. 1)
must be able to capture the real world concepts (guideline of correctness) and, second, it must be clear and intuitive in its use (guideline of clarity, see [BRU00], p. 33).

One the one hand, existing temporal modelling languages with their formal focus do not account for these special needs in the conceptual analysis phase. Modelling techniques successfully used for business process modelling are event-driven process chains (see [KNS92]) and activity diagrams (see [OM03a], p. 2-170; [Oe03], pp. 175). However, these languages, on the other hand, do not account for temporal aspects in business processes without modification and thus conflict with the guideline of correctness.

Modelling concepts, which capture the special temporal needs in business process models are not yet well developed. HÖHISEL, for example, models temporal aspects in business processes with the help of the Event Condition Action Alternative Action (ECAA) concept (see [Ho99]). His approach focuses on temporal events. We show in this paper that this is only one of four temporal aspects, which need to be considered. Furthermore, the ECAA notation gets overcomplicated as the complexity of the underlying process model increases (see [Ho99], p. 17, figure 19). The need of the decomposition of the ECAA-notation, however, is not addressed by the author. This complexity conflicts with the guideline of clarity.

The remainder of this paper is structured as follows. We analyse the temporal aspects of business processes from a theoretical point of view in section 2. In section 3 we design our solution to two temporal aspects, the periodicity of business processes and the temporal reference of information objects. The section finishes with the discussion of the limitations of our approach. New insights into temporal modelling have been developed in the UML field (see [ST99]; [Do99]; [KW02]; [APR02]; [FM02]). Consequently, in section 4 we discuss briefly how our approach can be implemented into the UML. Since these aspects are especially relevant in the E-Government domain we will provide an example of this domain. Section 5 summarises the paper and states the remaining problems.

2 Analysis

“A process is … a specific ordering of work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs: a structure for action.” (Da93, p. 5) According to this definition a process consists of at least one activity. Consequently, a coherent subset of the process’ activities is called sub-process. In business process modelling, however, processes are not modelled directly but only process types. Moreover, each activity modelled within a process model can represent a folded sub-process.

5 The only way to model temporal aspects in activity diagrams is by using the signal concept. However, this concept supports time events only (see [Oe03], p. 185). The new UML 2.0 concept time event focuses on events as well and disregards the relation of this event to the underlying business process.
Processes, especially in the E-Government domain, are always embedded in a particular temporal framework. Process instances are usually repeated within a specific period of time. The main period in the government processes is the fiscal period. The core processes are repeatedly executed for each fiscal year. However, sub-processes may have a different time period and may also be repeated within a sub-period of the fiscal year. These temporal aspects are either determined by law or by the characteristic of the processes themselves. For instance, the budget planning is executed for each fiscal period whereas the expenditures and revenues are coordinated each day.

Consequently, one of the temporal characteristics, which has to be addressed, is the **periodicity** in process models describing the temporal period in which the process is regularly repeated. Additionally, to each work activity **temporal constraints** can be assigned. For instance, the activities could be underpinned by durations or time limits. All these information can be used for business process scheduling.

Moreover, processes are based on information objects. These objects are either used as information source (defined input) to execute the process or they are altered by these processes (defined output; see [Sc98], p. 100). Information objects also have a **temporal reference** with which they are bound to the underlying business process instance. The creation of a monthly statistics (process), for example, may summarise all pre-existing weekly statistics (information objects).

Furthermore, not only the information object itself can be assigned to a specific time period but also its attributes. In other words, the **temporal state** of the attribute’s values might be interesting. The birthday of an employee, for example, is a constant whereas his or her salary may vary over the time. This history of the salary might be of special interest.

To sum up the results so far, at least four different temporal aspects must be regarded in business process modelling, the **periodicity** of business processes, **temporal constraints** of activities, the **temporal reference** of information objects and the **temporal state** of their attributes. However, the temporal constraints of work activities belong to the operations research discipline and have already been addressed (see [Jo74]; [Ru99], pp. 180). Moreover, the temporal state of the attributes of information objects has already been investigated in the field of temporal databases (see [TLW91]) and is particularly important for conceptual data modelling (see [KW02]). For these reasons and because we

---

6 In contrast to SCHÜTTE, who defines one information object for each process (see. [Sc98], p. 100), we use the possibility to define more than one information object for one process. SCHÜTTE’s constraint is thus regarded as special case since the business processes and the information objects must be of equal granularity.

7 Timing diagrams in UML 2.0 focus on the state change of objects only and thus focus on the temporal states of the object’s attributes.
concentrate explicitly on the process view, we will focus on the periodicity of business processes and the temporal reference of information objects.\(^8\)

Consequently, the modelling language in question must, firstly, provide constructs to express how often and in which temporal interval business process instances must be repeated. Secondly, the language must have an attribute of the information object construct, which expresses the temporal reference of this information object. Both aspects are further designed and formalised in the next section.

3 Design

3.1 Periodicity of Business Processes

In this subsection we formalise the periodicity of process models. Let \( t_p \) be defined as a temporal period. This period may not be bound to any specific data (e.g. December 2003) and is expressed as:

- \( y \equiv \text{year} \)
- \( m \equiv \text{month} \)
- \( w \equiv \text{week} \)
- \( d \equiv \text{day} \)

As shown before the core business process in the government domain is assigned to the period \( t_p = y \) because of the high importance of the fiscal year. Activities and subprocesses included in this core process, however, might be executed within different temporal periods.

The defined temporal periods are included one in another. This means that the temporal period day is included in the period week is included in the period month is included in the period year. This definition implies that, in a hierarchy of processes, the subprocesses may only have the same or a lower temporal period than the super-process. This regulation excludes, for instance, the possibility to model an activity with the temporal period \( t_p = y \) within a process with the period \( t_p = w \). This modelling is inconsis-

\(^8\) The consequences of these decision are the impossibility of our modelling language to capture temporal constraints and temporal states. Temporal aspects such as public holidays, time events and time intervals are not in the scope of this paper.

\(^9\) BRAUN ET AL. define an additional category \( s \) for each period, which does not fit into the categories year, month, week or day (see [BEG04], p. 65). This additional category enhances the possibility of expressing temporal constraints in process models but also the complexity of the constraints for this category. For simplicity we restrict our argumentation to the above mentioned four categories.
tent since it means that the process can never be finished within a week \((tp=w)\) because one of its activities finishes within a year \((tp=y)\).

3.2 Temporal Reference of Information Objects

To formalise the temporal aspects of the information objects needed by a specific business process these information objects are additionally assigned with a temporal reference relating them to the underlying process. So, each information object carries a state-property (by definition) and an additional temporal property (temporal reference).

The state (e. g. [acknowledged]) of an information object is defined by a specific set of values of the object’s attributes (see [Sc98], p. 102; see [OM03a], p. 2-146; see [CY91], p. 145).

The temporal reference \(tr\) of this information object is composed of a temporal period \(tp\) and a temporal constant \(tc\) \((tr=tp+tc)\). The temporal period is identical for each information object of a class and is a constant for this information object.

The temporal period \(tp\) has already been defined in subsection 3.1 and must be one of \(tp \in \{y; m; w; d\}\). The relations between the defined temporal periods imply for instance that an information object assigned to a monthly period is automatically associated with a specific year if it belongs to a sub process and its super process is assigned to a yearly time period. The conversion, however, does not hold.

The value of the time constant \(tc \in \mathbb{Z}\) defines a relative shift according to the temporal period. It is determined during the modelling process and only valid within this specific context. The time constant can be used to express that an information object belongs to the last or the next business process instance.

The temporal reference of an information object expresses two things. First, it defines to which temporal period it belongs to (year, month, week, day). Second, the time constant expresses to which temporal period it is assigned relatively to the underlying business process instance (last month, next week, last year).

In most cases the temporal period is identical with the period of the underlying process instance, thus \(tc=0\). A negative time constant results, if, for example, the process create statistics is mainly based on the equally named information object which resulted one month before. On the other hand, positive time constants need to be modelled, if information objects are created and/or prepared in the period of the underlying process instance but are logically used in the next period. An example is a plan (information object) prepared in the period of the business process instance in focus, which logically belongs to a time period in the future.
More formal, the temporal reference of an information object can be described as:

\[ tr = tp + tc \text{ with } tp \in \{ y; m; w; d \} \text{ and } tc \in \{ 0; \ldots; \infty \} \]

\[ kc \in \{ 0; \ldots; 12 \}; \quad tp = y \]

\[ kc \in \{ 0; \ldots; 5 \}; \quad tp = w \]

\[ kc \in \{ 0; \ldots; 7 \}; \quad tp = d \]

For consistency reasons it is necessary to assure, given \( tc < 0 \), that the information object is assigned to its specific state before it is used in the business process. Furthermore, given \( tc > 0 \), it is necessary to assure that the state of the information object used in the future process instance is equal to one of the states assigned to this information object of the last process instance.

### 4 Implementation

To demonstrate the significance of our argumentation for current modelling techniques, this section illustrates how our results can be implemented into UML. As shown earlier UML activity graphs lack the possibility to model the relevant temporal aspects. At this point we concentrate on the practical issues of this integration. **Braun et al.** have described how this integration can be formalised with a concrete UML profile and specified the constraints needed for assuring the model consistency (see [BEG04], pp. 70).

#### 4.1 Business Processes in UML

We use the multiple transition concept from **Oesterreich** to model the periodicity of business processes (see [Oe01], p. 189). The main idea behind this concept is the concurrency of processes. Usually this is visualised within an activity diagram using the fork and the join pseudostates.

**Oesterreich** transformed this idea to the instance level. The following example may illustrate this idea (see figure 1). The statement of a bank account consists of a set of different payment positions. Each of them documents a single payment. A process might check each of these payments separately. Intuitively, we would model this aspect with a loop running through all payment positions (figure 1 left).

The activity check payment position can be thought as a type for the account’s statement and the positions as instances of this process. Hence, the activity check payment position is executed for each position of this statement. In other words it

---

10 This formalisation shows also the limit of our approach. The restriction prohibits the modelling of any temporal reference. It is, for example, impossible to model an information object, which was created twenty months before \( (n-20) \).

---
can be thought as a fork of the process at the instance level (one concurrent process instance for each position) and a synchronisation of this fork.

Since this differentiation is made on the instance level we model the `check payment position` activity only once. The repetition is expressed by a specific constraint assigned to the two pseudostates. The constraint for each starts the multiple transition, e.g. creates a (fixed) number of instances of the following activities. These activities are synchronised by the subsequent join constraint all.

![Diagram](image)

**Figure 1:** The multiple transition concept according to OESTERREICH (see [Oe01], p. 189)

The idea of multiple transitions can be used similarly to model temporal aspects of business processes. The activities executed in sub periods can be thought as repeatedly executed instances according to the surrounding activities.

Consequently, the fork pseudostates are assigned with the constraints `each year`, `each month`, `each day` and the corresponding joins are constrained by `every year`, `every month` and `every day` respectively. These additional constraints must follow the rules documented in subsection 3.1.

### 4.2 Information Objects

The temporal reference of an information object is added as a specific property to this object in the form temporal period + time constant (tp+tc). The new property is visualised between the object’s class and the object’s state.

### 4.3 Example

We provide the following example to illustrate the concepts constructed in this paper and to demonstrate the constraints these concepts must fulfil.
In the example there is a business process create statistics (see figure 2, left), which is further analysed and unfolded (see figure 2, right). The process has the periodicity of the fiscal year. Hence, it is executed exactly once within one year.

The activities of the sub-process have a different periodicity. The first activities create monthly statistics and check monthly statistics are repeatedly executed within one month. This temporal shift is modelled with the fork and the join pseudostates and the additional constraints each month and every month respectively. Thus, these two activities are repeated twelve times within one business process cycle.

The statistics of the year before (with respect to the process instance at hand) is used for the creation of the yearly statistics (temporal reference \( y-1 \)). Consequently, the newly
created information object yearly statistics has the temporal reference $y$ since it is bound to the current process instance. At the end of the process the information object is in the same state ([approved]) as the required object at the process’ beginning.

Furthermore, the example visualises that all monthly created statistics and the yearly created statistics of the year before (according to the process instance) are needed to lastly create the yearly statistics.

This simple example illustrates the need of acknowledging temporal aspects in business process modelling. It demonstrates also the increasing complexity of the notation. This complexity can only be handled with the help of supporting tools. This functionality can be provided by any meta case tool, e.g. the Ecube Toolset developed at the Dresden University of Technology (see [Gr04]).

5 Summary

In section 2 we have showed, which temporal aspects need to be addressed when modelling business processes. Two temporal aspects are especially important. First, the periodicity of the processes themselves and, second, the temporal reference of the information objects. These core concepts are domain and language independent. This means that the concepts are possibly useful for other domains and can be implemented in other modelling languages (e.g. event-driven process chains) in a similar manner.

Our approach is based on semantic rich concepts. These concepts support a higher clarity in business process models for domain experts than alternative approaches. Furthermore, they are the basis for their tool-support. Lastly, the implementation of the proposed concepts in UML showed the practical applicability of these concepts.

Further research should include the implementation of the constraints in a CASE tool to minimize the effort for assuring the quality of the resulting models (guideline of economic efficiency, see [BRU00], p. 33). Furthermore, the temporal aspects addressed here should be implemented in other modelling language such as the Event-Driven Process Chain and the Business Process Modelling Language.

References


