EVENT-BASED INFORMATION SYSTEM MODELS

Lars Bækgaard
Aarhus School of Business, Aarhus University, Fuglesangs Alle 4, DK-8210 Aarhus V, Denmark

Keywords: Information Systems, modeling languages, activity models, events.

Abstract: Activity modeling plays important roles in information systems development. Activity models are used to support analysis and design of information systems. Events are central in modeling languages like EPC and BPMN that use events to control the order in which sub activities are performed. The purpose of this paper is to analyse a set of activity modeling languages with respect to their use of events as modeling constructs. We base our analysis on modeling examples and a conceptual model of events. We argue that it is possible to unify the languages' event concepts and to use a unified event concept to improve the languages.

1 INTRODUCTION

Activity modeling plays important roles in information systems development. Activity modeling and activity modeling languages have been the subject of significant research efforts lately (Kalnins, Barzdins et al. 2000; Lauesen 2003; Rittgen 2003; Andersen 2006; Lübke, Lüecke et al. 2006).

Events and event-based activity modeling are central in a number of modeling approaches (Keller and Teufel 1998; Bækgaard 1999; White 2004; Dietz 2006). However, well-known modeling approaches like EPC (Keller and Teufel 1998) and BPMN (White 2004) use events in different and apparently incompatible ways.

The purpose of this paper is to analyse a set of modeling languages and to argue that it is possible to use a conceptual event model to unify and improve the languages' use of events.

The paper is structured as follows. In Section 2 we discuss the notion of events and present a conceptual event model. In Section 3 we discuss and analyse three event-based activity modeling languages. In Section 4 we conclude the paper and suggest directions for future research.

2 EVENTS

In this section we discuss our understanding of the notion of events and we present a conceptual event model that we use in Section 3 to analyse event-based activity modeling languages.

We view an event as a significant occurrence whose duration is assumed to be negligible (Jackson 1983). Even the smallest process takes time. When we view a phenomenon as an event we assume that a further subdivision of the phenomenon is irrelevant in the given context and that it makes sense to neglect its duration. Events occur within processes like chemical processes, biological processes, technical processes, or social processes.

We can view phenomena like "Customer files an order", "Borrower borrows a book", "Website user clicks", "Heart beats", or "Elevator starts moving" as events. These phenomena have durations but in certain contexts it makes sense to assume that they occur at points in time.

Events may have descriptive properties and participants (Bækgaard 1999; Bækgaard 2004). All events have occurrence times. For example, ordering events may have properties like quantities and occurrence times. Participants can be human actors, IT-systems, machines, things like produced items, or information like sales reports. An event is shared if it has two or more participants. An ordering event can be viewed as a time point where a customer and a product meet. Ordering events may have participants like customers and products.

Events have consequences. They can trigger, terminate, and prohibit activities. When an event occurs activity may be initiated. When a book is borrowed in a library an activity that monitors timely returning of the book may be initiated. When

Bækgaard L. (2007).
EVENT-BASED INFORMATION SYSTEM MODELS.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - ISAS*, pages 587-590
DOI: 10.5220/0002390305870590
Copyright © SciTePress

a business receives an order one or more employees may carry out packing and shipping activities, and an ERP system may carry out invoicing activities. Events may be constrained in the sense that participants may not be ready to participate in requested events at certain points in time. For example, a book that is currently borrowed can not participate in a borrowing event.

Events can be viewed as information objects that are observed and described. For example, a data warehouse with multi-dimensional information about customer behaviour may be based on information about events like ordering and payment (Bækgaard 1999). Information about events can be registered, stored, manipulated, and presented for actors. Information about an event can refer to its type, participants, and properties.

Individual events can be grouped into types of events with similar types of participants and properties. We use event expressions to represent individual events as illustrated by the following example: *Borrow [X: Borrower, Y: BookItem] (September 29 2001)*.

This expression states that an event named Borrow has occurred with the participants X and Y. X belongs to the set Borrower and Y belongs to the set BookItem. The event has the property "September 29 2001". We can interpret an event expression in different ways. The example may represent the fact that the Borrower element X and the BookItem element Y have participated in a Borrow event on September 29 2001.

We use event signatures on the following form to define types of conforming events: *Name [Participants] (Properties)*.

The element Participant defines the types of participants that can participate in events of the defined type. The element Property defines the types of properties of events of the defined type.

An event expression conforms to a signature if it has the same name as the signature and if its participants and properties conform to the participant and property types of the signature. For example, the event expression Borrow [Borrower1, BookItem4] (September 29, 2001) conforms to the signature Borrow [Borrower, BookItem] (Date).

3 MODELING LANGUAGES

In this section we discuss and analyse three event-based activity modeling languages. We use modeling examples and our conceptual event model from Section 2 to support our analysis.

3.1 EPC Diagrams

Event-driven Process Chains (EPC) use events, functions, and logical operators to represent business activities. An event represents the entering of a state that causes a consequence a state change (Keller and Teufel 1998). Events trigger functions and/or they are the results of functions.

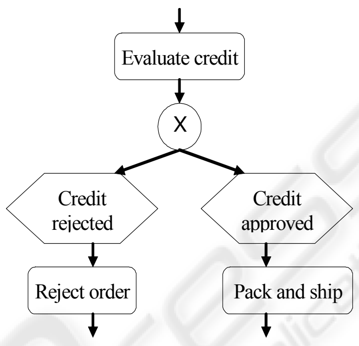

Figure 1: Partial EPC diagram.

Events can be used to control the flow of functions as illustrated in the partial diagram in Figure 1 that illustrates how events are used to trigger functions. First, the function "Evaluate credit" is executed. Then, one of two options is chosen. If the event "Credit rejected" occurs the function "Reject order" is executed. If the event "Credit approved" occurs the function "Pack and ship" is executed.

An EPC event can be represented by its name in terms of a signature on the form *Name [] ()*. EPC-diagrams do not utilize the fact that events can have multiple interacting participants. The selection of a the particular sequence depends on the state changes (represented by events) that occur. Because of the fact that EPC-events has no participants or properties EPC is based on the simplest view on events—namely a named state change.

3.2 BPMN Diagrams

Business Process Modeling Notation (BPMN) supports business activity modeling in terms of events and activities. BPMN is based on the assumption that an event is something that happens during the course of a business process. Events affect the flow of a process and usually have a cause or an impact. BPMN has restricted the use of events to include only those types of events that will affect the sequence or timing of activities of a process (White 2004).

Events can be used to control the flow of activities as illustrated by the partial diagram in Figure 2 that illustrates how events are used to trigger and interrupt activities. First, the activity "Evaluate credit" is executed. Then, one of two options is chosen. If the event "Credit rejected" occurs the activity "Reject order" is executed. If the event "Credit approved" occurs the activity "Pack and ship" is executed. The activity "Pack and ship" can be interrupted by the event "E".

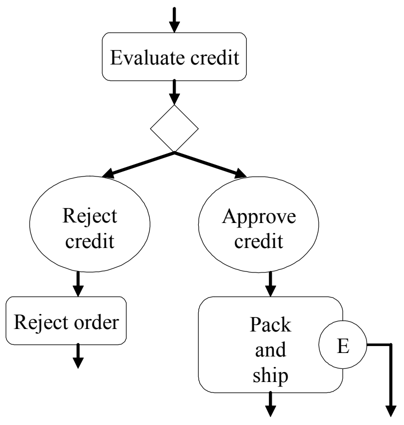

Figure 2: Partial BPMN diagram.

An event may have an associated occurrence time. No actors or objects are involved. This implies that a BPMN event can be represented by its name and occurrence time in terms of a signature on the form *Name [] (Time)*. Consequently, events cannot be used to synchronize the activities of multiple interacting participants.

3.3 Event-activity Diagrams

Event-activity diagrams are UML activity diagrams extended with shared events (Bækgaard 2004). An event-activity diagram defines a type of activity and it is composed of triggering events, interrupting events, and an activity. An activity is initiated when a triggering event occurs. The activity runs until it reaches an exit point or until an interrupting event occurs. One or more actors carry out the activity.

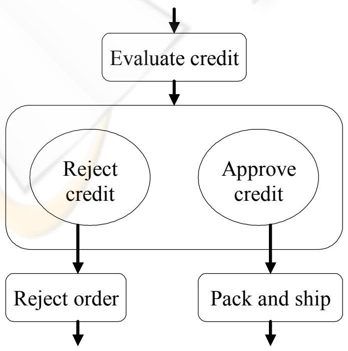

Figure 3: Partial Event-activity diagram.

Events can be used to control the flow of activities as illustrated by the partial diagram in Figure 3 that illustrates how events are used to trigger activities. First, the activity "Evaluate credit" is executed. Then, one of two options is chosen. If the event "Credit rejected" occurs the activity "Reject order" is executed. If the event "Credit approved" occurs the activity "Pack and ship" is executed.

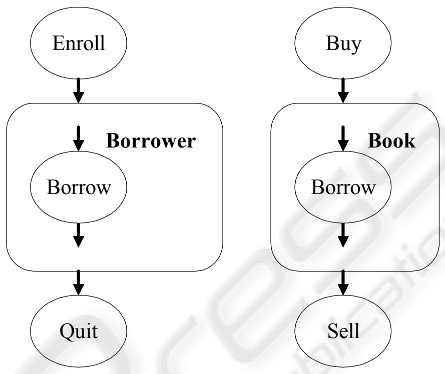

Figure 4: Partial Event-activity diagrams.

Events can be used to trigger and interrupt activities and to synchronize two or more autonomous activities as illustrated by the partial diagrams in Figure 4. An event-activity diagram defines an activity type that can have multiple, concurrent instances. An instance of an activity type is generated by the occurrence of an event that conforms to a triggering event type. An instance of the activity "Borrower" is triggered by an "Enroll" event and it is interrupted by a "Quit" event. An instance of the activity "Book" is triggered by a "Buy" event and it is interrupted by a "Sell" event. Each "Borrow" events has two participants—a borrower and a book. This implies that each "Borrow" event synchronizes the activities of the participating borrower and book.

Events with multiple participants represent time points where multiple activities must be synchronized.

Event-activity diagrams are based on a view on events that corresponds to a general signature on the form *Name [Participants] (Properties)*.

3.4 Discussion

EPC diagrams, BPMN diagrams, and Event-activity diagrams use rather similar modeling constructs to use events to trigger activity as indicated by the previous examples—the partial diagrams in Figure 1, Figure 2, and Figure 3 have the same structure.

BPMN diagrams and Event-activity diagrams use events to interrupt activities. EPC diagrams do not support the modeling of interrupting events. In a

BPMN diagram all activities can have an attached interrupting event that defines conditions for interruption of the activity. An Event-activity diagram can have a set of attached event types that defines interrupting events. However, the individual activities in Event-activity diagrams cannot have attached interrupt events. EPC diagrams and Event-activity diagrams can be improved by incorporating the interrupt mechanism that us used in BPMN.

Event-activity diagrams are based on events with multiple participants. Therefore, events can be used to synchronize two or more activities each of which are defined by an event-activity diagram as illustrated by the partial diagrams in Figure 4 that are synchronized when the shared event “Borrow” occurs.

The notion of shared events with multiple participants can be used to improve the modeling power of EPC diagrams and BPMN diagrams. This could be expressed in terms of event signatures in the following way. The general signature of EPC events would become *Name [Participants] ()* and the general signature of BPMN diagrams would become *Name [Participants] (Time)*.

The consequence of this is that EPC diagrams and BPMN diagrams will be able to use shared events to synchronize two or more activities in the same way that Event-activity diagrams are (see Figure 4).

4 CONCLUSION

Our analysis of three event-based activity modeling languages has revealed that their seemingly incompatible event concepts are special cases of a more general event concept.

Most importantly, we have argued that the event concepts of EPC and BPMN can be extended to support shared events with multiple participants. We have illustrated elsewhere that shared events make it possible to support modeling of multiple instances of activities of the same type (Bækgaard 2004).

Event-activity diagrams are based on events with participants. This perspective on events is different from the event perspective used by EPC (Dehnert 2002) and BPMN (White 2004). These languages do not consider the participants of events.

Event-activity diagrams use events with participants to synchronize two or more concurrent activities. The subdivision of activities into concurrent, interacting activities that is facilitated by this mechanism makes it possible to define activities in a distributed manner that resembles the distribution of activity among actors in a business.

Future work includes design of a modeling language that is based purely on a general event signature on the form *Name [Participants] (Properties)*. The purpose is twofold. First, the purpose is to illustrate the modeling power of being able to use shared events to synchronize two or more independent activities. Second, the purpose is to define the semantics that are necessary in order to incorporate synchronization based on shared events into modeling languages like EPC and BPMN.

REFERENCES

Andersen, P. B. (2006). "Activity-Based Design of Information Systems". *European Journal of Information Systems* 15: 9-25.

Bækgaard, L. (1999). "Event-Entity-Relationship Modeling in Data Warehouse Environments". *DOLAP'99. International Workshop on Data Warehousing and OLAP*. Kansas City, USA.

Bækgaard, L. (2004). "Event-Based Activity Modeling". *ALOIS'04 - Action in Language, Organisation and Information Systems*. Linköping, Sweden.

Dehnert, J. (2002). "Making EPCs fit for Workflow Management". *EPK'2002 - Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten*. Trier, Germany.

Dietz, J. L. G. (2006). *Enterprise Ontology. Theory and Methodology*, Springer-Verlag.

Jackson, M. (1983). *System Development*, Prentice Hall International.

Kalnins, A., J. Barzdins, et al. (2000). "Modelling Languages and Tools: State of the Art". *2nd International Conference on Simulation, Gaming, Training and Business Process Reengineering*. Riga: 211-214.

Keller, G. and T. Teufel (1998). *SAP R/3 Process-Oriented Implementation*, Addison-Wesley Longman Limited.

Lauesen, S. (2003). "Task Descriptions as Functional Requirements." *IEEE Software*.

Lübke, D., T. Lüecke, et al. (2006). "Using Event-Driven Process Chains for Model-Driven Development of Business Applications". *Workshop XML4BPM 2006 - XML Integration and Transformation for Business Process Management*. Passau, Germany.

Rittgen, P. (2003). "Business Processes in UML". *UML and the Unified Process*. L. Favre, Idea Group Publishing.

White, S. A. (2004). "Introduction to BPMN", WWW.BPMN.ORG