Chapter XV
Foundations of Business Process Modeling

Jan Mendling
Queensland University of Technology, Australia

ABSTRACT

This chapter provides an overview of business process management and business process modeling. We approach business process management by giving a historical classification of seminal work, and define it by the help of the business process management life cycle. Business process models play an important role in this life cycle, in particular, if information systems are used for executing processes. We deduct a definition for business process modeling based on a discussion of modeling from a general information systems point of view. In the following, we detail business process modeling techniques, in particular, modeling languages and modeling procedures for business process modeling. Finally, we discuss some future trends with a focus on the business process execution language for Web services (BPEL), and conclude the chapter with a summary. The chapter aims to cover business process modeling in a comprehensive way such that academics and practitioners can use it as a reference for identifying more specialized works.

INTRODUCTION

This section provides an overview of business process management. The first part elaborates on the background of business process management by giving a historical classification of seminal work. The second part defines business process management and illustrates it by the help of the business process management life cycle. Business process models play an important role in this life cycle.

HISTORY OF BUSINESS PROCESS MANAGEMENT

In the last couple of years, there has been a growing interest in business process management, from practice as well as from business administration and information systems research. In essence, business process management deals with the efficient coordination of business activities within and between companies. As such, it can be related to several seminal works on economics and business administration. Fayol (1966), as one of the founders of modern organization theory recommended a subdivision of labor in order to increase productivity. Adam Smith (1776) illustrated its potential benefits by analyzing pin production. As a drawback, subdivision of labor requires coordination between the subtasks. Business process management is concerned with coordination mechanisms, in order to leverage the efficient creation of goods and services in a production system based on such subdivision of labor. In this context, the individual tasks and the coordination between them can be subject to optimization efforts. Frederick Taylor advocated the creation of an optimal work environment based on scientific methods to leverage the most efficient way of performing individual work steps. In the optimization of each step, he proposed to "select the quickest way," to "eliminate all false movements, slow movements, and useless movements," and to "collect into one series the quickest and best movements" (Taylor, 1911). The efficient coordination of business processes is addressed by the innovation of the assembly line system. Its inventor Ford (1926), proudly praised the production cycle of only 81 hours in his company "from the mine to the finished machine" to illustrate the efficiency of the concept.

In academia, Nordsieck was one of the first to distinguish structural and process organization (Nordsieck, 1932, 1934). He described several types of workflow diagrams, for example, for subdivision and distribution of labor, sequencing of activities, or task assignment (Nordsieck, 1932). In this context, he identifies the order of work steps and the temporal relationship of tasks as the subject of process analysis with the overall goal of integrating these steps. He distinguishes between five levels of automation: free course of work, concerning the contents bound course of work, concerning the order bound course of work, temporally bound course of work, and concerning the beat bound course of work (Nordsieck, 1934).

In the decades after World War II, operations research devoted more attention to structural organization than to process organization. In the early 1970s, it became apparent that information systems would become a new design dimension in an organizational setting (see Grochla & Szyperski, 1975). But the focus, even in this context, remained on the structure. At that time, the logic of business processes used to be hard-coded in applications such as production floor automation systems and was, therefore, difficult to change (Hsu & Kleissner, 1996; zur Muehlen, 2004). Office automation technology during the late 1970s was the starting point for a more explicit control over the flow of information and the coordination of tasks. The basic idea was to build electronic forms for clerical work that was originally handled via paper. In his doctoral thesis, Zisman (1978, 1977) used Petri nets (Petri, 1962a, 1962b) to specify the clerical work steps of an office agent and introduced a respective prototype system called SCOOP. A comparable approach was presented by Ellis (1979), who modelled office procedures as information control nets, a special kind of Petri nets consisting of activities, precedence constraints, and information repositories. An overview of further work on office automation is provided in Ellis and Nutt (1980).

Although the business importance of processes received some attention in the 1980s (see Porter, 1985) and new innovations were introduced in information system support of processes, for instance (e.g., system support for communication processes (Winograd, 1987-1988) based on speech act theory (Austin, 1962; Searle, 1969)), it was only in the early 1990s that workflow management prevailed as a new technology to support business processes. An increasing number of commercial vendors of workflow management systems benefited from new business administration concepts and ideas such as process innovation (Davenport, 1993) and business process reengineering (Hammer & Champy, 1993). On the other hand, these business programs heavily relied on information system technology, in particular workflow systems, in order to establish new and more efficient ways of doing business. In the 1990s, the application of workflow systems, in particular, those supporting information systems integration processes, profited from open communication standards and distributed systems technology that both contributed to interoperability with other systems (see Georgakopoulos, Hornick, & Sheth, 1995). The workflow management coalition (WfMC) founded in 1993 is of special importance for this improvement (Hollingsworth, 2004). The historical overview of office automation and workflow systems given in zur Muehlen (2004) nicely illustrates this breakthrough. This period also saw a growing body of scientific publications on workflow technology and process specification (see van der Aalst, 1998; Casati, Ceri, Pernici, & Pozzi, 1995; Ellis & Nutt, 1993; Georgakopoulos et al., 1995; Jablonski & Bussler, 1996; Leymann & Roller, 2000; Oberweis, 1996; Oberweis & Sander, 1996; Reichert & Dadam, 1998; Scholz-Reiter & Stickel, 1996). Up to the late 1990s, intra-enterprise processes remained the major focus of business process management (see Dayal, Hsu, & Ladin, 2001).

Since the advent of the extended markup language (XML) and Web services technology, application scenarios for business process integration have become much easier to implement in an inter-enterprise setting. Current standardization efforts mainly address interoperability issues related to such scenarios (see Mendling, Nüttgens, & Neumann, 2004; Mendling, zur Muehlen, & Price, 2005; Mendling, Neumann, & Nüttgens, 2005). The common interest of the industry to facilitate the integration of inter-organizational processes leverages the specification of standards for Web service composition, like the business process execution language for Web services (Alves et al., 2007, Andrews et al., 2003, Curbera et al., 2002), for Web service choreography, like the Web service choreography description language (WS-CDL) (Kavantzas et al., 2005), or for inter-organizational processes based on ebXML and related standards (see Hofreiter, Huemer, & Kim, 2006). The integration of composition and choreography languages is currently one of the main research topics in this area (see Mendling & Hafner, 2005; Weber, Haller, & Mülle, 2006).

Today, business process management is an important research area that combines insights from business administration, organization theory, computer science, and computer supported cooperative work. Furthermore, it is a considerable market for software vendors, IT service providers, and business consultants.

Definition of Business Process Management

Since the beginnings of organization theory, several definitions for business processes have been proposed. Nordsieck, in the early 1930s, describes a business process as a sequence of activities producing an output. In this context, an activity is the smallest separable unit of work performed by a work subject (Nordsieck, 1934). In this tradition, Becker and Kugeler (2003) propose the following definition:

A process is a completely closed, timely and logical sequence of activities which are required to work on a process-oriented business object. Such a process-oriented object can be, for example, an invoice, a purchase order or a specimen. A business process is a special process that is directed by the business objectives of a company and by the business environment. Essential features of a business process are interfaces to the business partners of the company (e.g., customers, suppliers).

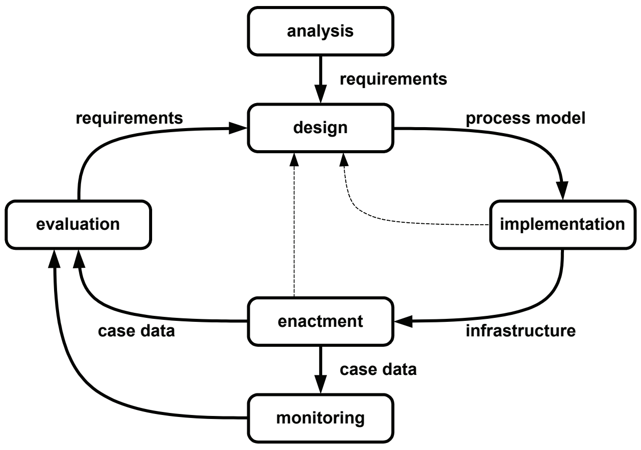
As Davenport (1993) puts it, a "process is thus a specific ordering of work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs: a structure for action." Van der Aalst and van Hee (2002) add that the order of the activities is determined by a set of conditions. In this context, it is important to distinguish between the business process and several individual cases. Consider a business process such as car production. This process produces cars as an output. The production of one individual car that is sold to customer John Smith is a case. Accordingly, each case can be distinguished from other cases, and a business process can be regarded as a class of similar cases (van der Aalst & van Hee, 2002).

Related to business processes and information systems support, several categorization schemes were proposed. As an extension of Porter's value chain model (see Porter, 1985), van der Aalst and van Hee (2002) distinguish between production, support, and managerial processes. Production processes create products and services of a company that are sold to customers. These processes are of paramount importance since they generate income for the company. Support processes establish an environment in which the production processes go smoothly. Therefore, they do not only include maintenance activities, but also marketing and finance. Managerial processes direct and coordinate production and support processes. They are basically concerned with defining goals, preconditions, and constraints for the other processes. Leymann and Roller (2000) provide a classification scheme[1] for processes based on their business value and their degree of repetition. They use the term production process to refer to those processes that have both a high business value and a high degree of repetition. Administrative processes are also highly repetitive, but of little business value. Furthermore, collaborative processes are highly valuable, but hardly repeatable. Finally, ad hoc processes are neither repetitive nor valuable. Leymann and Roller conclude that information systems support should focus on production processes. In particular, workflow management systems are discussed as a suitable tool. Further classifications can be found, for example, in Dumas, ter Hofstede, and van der Aalst (2005).

Business process management can be defined as the set of all management activities related to business processes. In essence, the management activities related to business processes can be idealistically arranged in a life cycle. Business process management life cycle models have been described for instance in van der Aalst and van Hee (2002), Dumas et al. (2005), and zur Muehlen (2004). In the remainder of this section, we mainly follow the life cycle proposed by zur Muehlen (2004), firstly, because it does not only include activities but also artefacts, and secondly, because it consolidates several earlier life cycle models for business process management. The life cycle shares the activities analysis, design, and implementation with the general process of information systems development identified by Wand and Weber (1990). Altogether, the life cycle comprises the management activities of analysis, design, implementation, enactment, monitoring, and evaluation. The solid arcs represent the typical order of these activities; while the dotted arcs depict atypical feedback loops (see Figure 1).

- **Analysis:** The business process management life cycle is entered with an analysis activity (see Figure 1). This analysis covers both the environment of the process and the organization structure. The output of this step is a set of requirements for the business process, such as performance goals.
- **Design:** These requirements drive the subsequent design activity. In particular, the design includes the identification of process activities, the definition of their order, the assignment of resources to activities, and the definition of the organization structure. These different aspects of process design are typically formalized as a business process model. This model can be tested in a simulation if it meets the design requirements.[2]
- **Implementation:** The process model is then taken as input for the implementation. In this phase, the infrastructure for the business process is set up. This includes (among others) training of staff, provision of a dedicated work infrastructure, or the technical implementation and configuration of software. If the process execution is to be supported by dedicated information systems, the process model is used as a blueprint for the implementation.
- **Enactment:** As soon as the implementation is completed, the actual enactment of the

Figure 1. Business process management life cycle



process can begin. In this phase, the dedicated infrastructure is used to handle individual cases covered by the business process. The enactment produces information such as consumption of time, resources, materials, and so forth, for each of the handled cases. This data can be used as input for two subsequent activities: monitoring and evaluation.

- **Monitoring:** A continuous activity that is performed with respect to each individual case. Depending on process metrics, as, for instance, maximum waiting time for a certain process activity, monitoring triggers respective counteractions if such a metric indicates a problematic situation.
- **Evaluation:** On the other hand, considers case data on an aggregated level. The performance results are compared with the original requirements and sources of further improvement are discussed. In this way, evaluation leads to new requirements that are taken as input in the next turn of the business process management life cycle.

The business process management life cycle reveals that business process models play an important role in the design, implementation, and enactment phase, especially when information systems support the process enactment. Thus, they are valuable resources for continuous process improvement, quality management, knowledge management, ERP system selection, and software implementation (Rosemann, 2003). Current market research supports this relevance, since about 90% of the participating companies in a survey conducted or considered business process modeling (Palmer, 2007). In practice, software tools play a decisive role in performing the various management activities in an efficient and effective manner. There are several commercial and academic tools which support different life cycle activities (van der Aalst & van Hee, 2002). In order to link these tools, the workflow management coalition has proposed five interfaces in a reference model (Hollingsworth, 1994). In particular, the availability of tools is important to the modeling of business processes in a correct and consistent way.

BACKGROUND ON BUSINESS PROCESS MODELING

Before defining business process modeling, the term "modeling" has to be discussed in a more general setting. Nordsieck (1932) has emphasized that "the utilization of symbols enables the model not only to replace or to complement natural language for the representation of complex matters, but to reveal the notion of the subject matter often in a more comprehensive way as with any other form of representation." The most protuberant features of a model are brevity, clarity, precision, and its graphic quality (Nordsieck, 1932). In the first part, we discuss the foundations of modeling. The second part introduces concepts related to business process modeling techniques.

Foundations of Modeling

Stachowiak (1973) defines a model as the result of a simplifying mapping from reality that serves a specific purpose. According to this perception, there are three important qualities a model should possess. Firstly, there is a mapping that establishes a representation of natural or artificial originals that can be models itself. Secondly, only those attributes of the original that are considered relevant are mapped to the model; the rest is skipped. Therefore, the model provides an abstraction in terms of a homomorphism in a mathematical sense (Kühne, 2006). Thirdly, the model is used by the modeller in place of the original at a certain point in time and for a certain purpose. This means that a model always involves pragmatics.

A weakness of Stachowiak's concept of a model is that it implies an epistemological position of positivism.[3] This is, for instance, criticized in Schütte and Rotthowe, where the authors propose an alternative position based on insights from critical realism and constructivism.[4] This position regards a model as a "result of a construct done by a modeller" (Schütte & Rotthowe, 1998). As such, it is heavily influenced by the subjective perception of the modeller. This fact makes modeling a non-deterministic task (Mendling & Recker, 2007), which requires standards in order to achieve a certain level of inter-subjectivity. The guidelines of modeling (GoM) (Becker, Rosemann, & Schütte, 1995; Becker, Rosemann, & von Uthmann, 2000; Schütte & Rotthowe, 1998) define principles that serve this standardization purpose. They are applicable for either epistemological positions or positivism and constructivism, because both the choice for a certain homomorphism (positivist position) and the perception of the modeller (constructivist position) introduce subjective elements.

Therefore, the guidelines of modeling (Becker et al., 1995; Schütte & Rotthowe, 1998) include six particular principles for achieving inter-subjectivity of models. The first three define necessary preconditions for the quality of models, that is, correctness, relevance, and economic efficiency, and the other three are optional, that is, clarity, comparability, and systematic design.

- **Correctness:** Firstly, a model has to be syntactically correct. This requirement demands the usage of allowed modeling primitives and combining them according to predefined rules. Secondly, a model must be semantically correct. Therefore, it has to be formally correct and consistent with the (perception of the) real world.
- **Relevance:** This criterion demands that only interesting parts of the universe of discourse are reflected in the model. It is, therefore, related to the notion of completeness as proposed in Batini, Lenzerini, and Navathe (1986).
- **Economic efficiency:** This guideline introduces a trade-off between benefits and costs of putting the other criteria into practice. For example, semantic correctness might be neglected to a certain extent, in case achieving it is too expensive.
- **Clarity:** This is a highly subjective guideline demanding that the model must be understood by the model user. It is primarily related to layout conventions or the complexity of the model.
- **Comparability:** Demands consistent utilization of a set of guidelines in a modeling project. Among others, it refers to naming conventions.

- **Systematic design:** This guideline demands a clear separation between models in different views (e.g., statical aspects and behavioral aspects) and defined mechanisms to integrate them.

An alternative, but more abstract framework for assessing the quality of modeling is the SEQUAL framework (Krogstie, Sindre, & Jørgensen, 2006; Lindland, Sindre, & Sølvberg, 1994). It builds on semiotic theory and defines several quality aspects based on relationships between a model, a body of knowledge, a domain, a modeling language, and the activities of learning, taking action, and modeling. In essence, syntactic quality relates to model and modeling language, semantic quality to model, domain, and knowledge, and pragmatic quality relates to model and modeling and its ability to enable learning and action. Although the framework does not provide an operational definition of how to determine the various degrees of quality, it has been found useful for business process modeling in experiments (Moody, Sindre, Brasethvik, & Sølvberg, 2002).

Beyond these quality considerations, there are several approaches that discuss different layers of modeling, and the relationship of the meta-layer towards philosophical theories. The following paragraph sketches ontology and meta-modeling as two alternative foundations. These two approaches are chosen as examples for their wide-spread application in information systems research.

Ontology is the study of being. It seeks to describe what is in the world in terms of entities, categories, and relationships. It is a prominent subdiscipline of philosophy. Wand and Weber were among the first to adopt ontology for a foundation of information systems modeling (see Wand & Weber, 1990, 1995). They make two basic assumptions. Firstly, as information systems reflect what is in the real world they should also be modelled with a language that is capable of representing real-world entities. Secondly, the ontology proposed by Bunge (1977) is a useful basis for describing the real world. The so-called Bunge-Wand-Weber (BWW) ontology proposed by Wand and Weber includes a set of things that can be observed in the world. They should be identified in the process of modeling a specific domain and fulfill certain consistency criteria (Wand & Weber, 1995). For examples of other ontological models, refer to Guizzardi, Herre, and Wagner (2002) and Wand and Weber (2002). Recently, ontology languages, such as OWL (McGuinness & Harmelen, 2004), have become popular for defining domain ontologies to be used as a component of the semantic Web (Berners-Lee, Hendler, & Lassila, 2001).

Metamodeling frees modeling from philosophical assumptions by extending the subject of the modeling process to the general (i.e., meta) level. The philosophical theory of this level, such as, for instance, an ontology, is replaced by a metamodel. The difference to an ontological foundation is that a metamodel does not claim any epistemological validity. Essentially, the metamodel identifies the abstract entities that can be used in the process of designing models, that is, in other words, the metamodel represents the modeling language (see Atkinson & Kühne, 2001b; Karagiannis & Kühn, 2002; Kühne, 2006). The flexibility gained from this meta-principle comes at the cost of relativism: as a metamodel is meta relative to a model, it is a model itself. Therefore, a metamodel can also be defined for the metamodel and it is called metametamodel. This regression can be continued to infinity without ever reaching an epistemological ground.[5] Most modeling frameworks define three or four modeling levels, see UML's meta object facility (OMG, 2002), CASE data interchange format (CDIF) (Flatscher, 1998), or graph exchange language (GXL) (Winter, Kullbach, & Riediger, 2001). The definition of a modeling language based on a metamodel is more often used than the explicit reference to a philosophical position. Examples of metamodeling can be found in Atkinson and Kühne, Atkinson and Kühne, Atkinson and Kühne, Österle and Gutzwiller, Scheer, and Scheer. Several tools like MetaEdit (Kelly Lyytinen, & Rossi, 1996;

Smolander, Lyytinen, Tahvanainen, & Marttiin, 1991), Protegé (Noy, Fergerson, & Musen, 2000), or ADONIS (Junginger, Kühn, Strobl, & Karagiannis, 2000) support metamodeling in such a way that modeling languages can be easily defined by the user.

The meta-hierarchy provides a means to distinguish different kinds of models. Still, a model can never be a metamodel by itself, but only relative to a model for which it defines the modeling language. Models can also be distinguished depending on the mapping mechanism (Strahringer, 1996): Non-linguistic models capture some real-world aspects as material artefacts or as pictures. Linguistic models can be representational, verbal, logistic, or mathematical. Models also serve diverse purposes. Focusing on business administration, Kosiol (1961) distinguishes descriptive models, explanatory models, and decision models. In this context, descriptive models capture objects of a certain area of discourse and represent them in a structured way. Beyond that, explanatory models define dependency relationships between nomological hypotheses. These serve as general laws to explain real-world phenomena, with a claim for empirical validity. Finally, decision models support the deduction of actions. This involves the availability of a description model to formalize the setting of the decision, a set of goals that constraint the design situation, and a set of decision parameters.

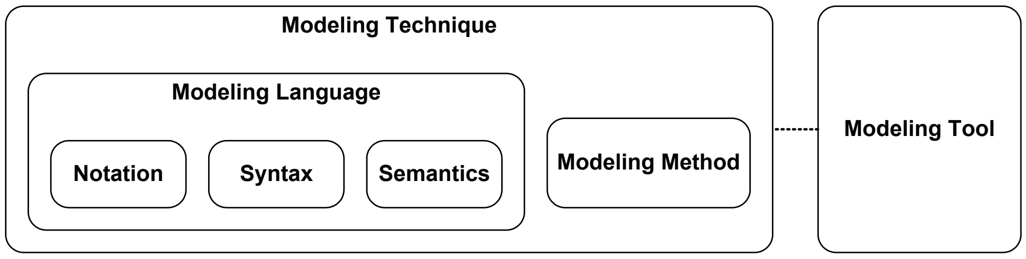
Business Process Modeling Techniques

The explicit definition of a modeling technique is a useful means to address several of the quality requirements that we discussed in the previous section. A modeling technique consists of two interrelated parts: a modeling language and a modeling method[6] (see Figure 2). The modeling language consists of three parts: syntax, semantics, and optionally, at least one notation. The syntax provides a set of constructs and a set of rules how these constructs can be combined. A synonym is modeling grammar (Wand & Weber, 2002, 1990, 1995). Semantics bind the constructs defined in the syntax to a meaning. This can be done in a mathematical way, for example, by using formal ontologies or operational semantics. The notation defines a set of graphical symbols that are utilized for the visualization of models (Karagiannis & Kühn, 2002). The modeling method defines procedures by which a modeling language can be used (Wand & Weber, 2002). The result of applying the modeling method is a model that complies with a specific modeling language.[7] Consider, for example, entity-relationship diagrams (ERDs) as defined by Chen (1976). Since they define a modeling language and a respective modeling method, ERDs are a modeling technique. Entities and relationships are syntax elements of its language. They are used to capture certain semantics of a universe of discourse. The notation represents entities as rectangles and relationships as arcs connecting such rectangles carrying a diamond in the middle. Respective procedures, like looking for nouns and verbs in documents, define the modeling method. In practice, modeling tools are of crucial importance for the application of a modeling technique. Among others, they support the specification of models, the redundancy controlled administration of models, multi-user collaboration, and model reuse via interfaces to other tools (Rosemann, 2003). A recent comparison of business process modeling tools is reported by Ami and Sommer (2007).

Against this background, the terms business process model, business process modeling language, and business process modeling can be defined as follows:

- A business process model is the result of mapping a business process. This business process can be either a real-world business process as perceived by a modeller, or a business process conceptualized by a modeller.
- Business process modeling is the human activity of creating a business process model. Business process modeling involves an abstraction from the real-world business process,

Figure 2. Concepts of a modeling technique



because it serves a certain modeling purpose. Therefore, only those aspects relevant to the modeling purpose are included in the process model.

- Business process modeling languages guide the procedure of business process modeling by offering a predefined set of elements and relationships for the modeling of business processes. A business process modeling language can be specified using a metamodel. In conjunction with a respective method, it establishes a business process modeling technique.

This definition requires some comments. In contrast to Stachowiak (1973), it does not claim that the business process model is an abstraction and serves a purpose. These attributions involve some problems about whether a model always has to be abstract or to serve a purpose. Instead, the procedure of business process modeling is characterized in such a way that it is guided by abstraction and a purpose in mind. This is important as a model is not just a "representation of a real-world system" as Wand and Weber (1990) put it, but a design artefact, in the sense of Hevner, March, Park, and Ram (2004), that itself becomes part of the real world as soon as it is created. Beyond that, business process models can be characterized as linguistic models that are mainly representational and mathematical. The representational aspect points to the visual notation of a business process modeling language, while the mathematical notion refers to the formal syntax and semantics. In practice, business process models are often used for documentation purposes (Davies, Green, Rosemann, Indulska, & Gallo, 2006). Therefore, they can be regarded as descriptive models for organization and information systems engineers. Still, they also serve as explanatory and decision models for the people who are involved in the actual processing of cases. In the following section, we will utilize the concepts related to modeling techniques as illustrated in Figure 2, for the introduction of different popular business process modeling languages.

BUSINESS PROCESS MODELING LANGUAGES AND METHODS

In this section, we focus on the control flow perspective of business process models and introduce several popular business process modeling languages. In particular, we introduce Petri nets, EPCs, YAWL, UML activity diagrams, and BPMN, and compare these languages based on the semantics of their routing elements and based on the workflow patterns defined by van der Aalst, ter Hofstede, Kiepuszewski, and Barros. Furthermore, we discuss how the process of modeling can be described as a method, and which measures of quality assurance should be considered.

Modeling Languages

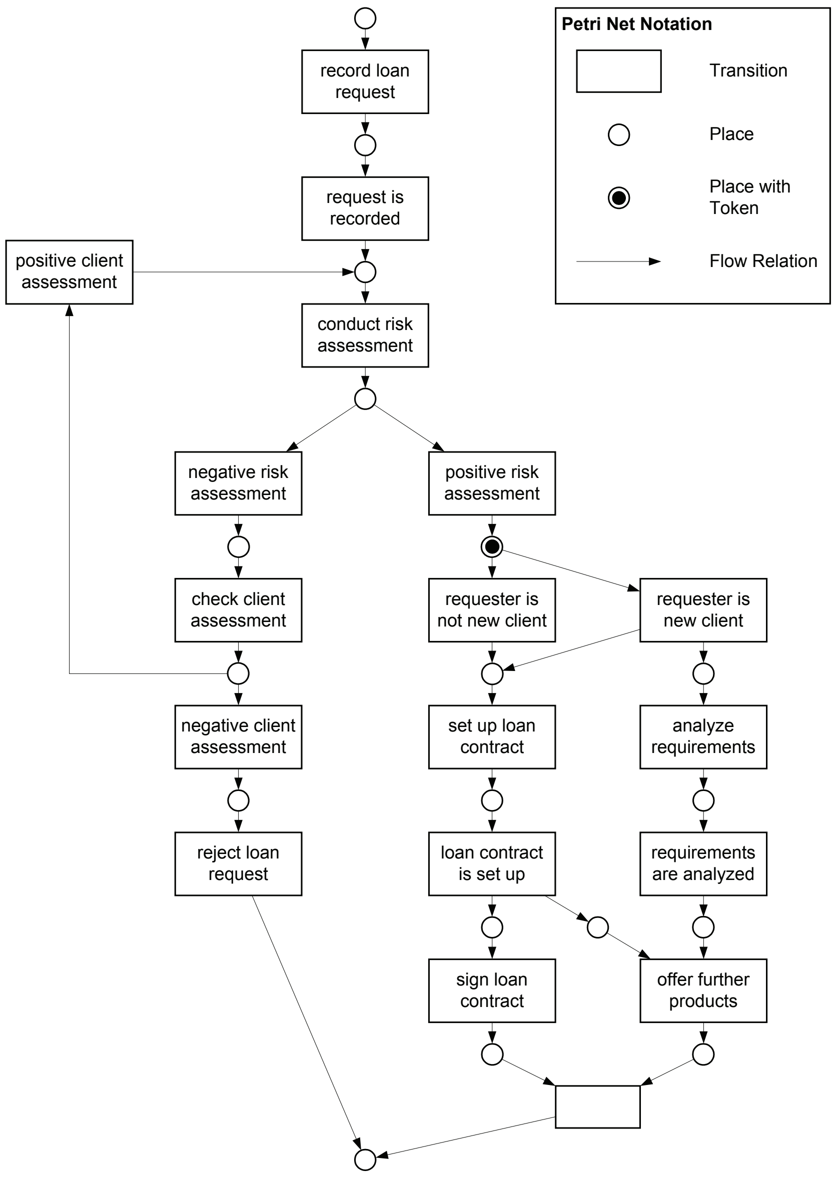
Petri Nets

Petri nets introduced by Petri (1962a, 1962b) are a formal language that can be used for the specification of business processes. The syntax of Petri nets is defined as a special kind of a graph with two different node types. So-called transitions typically represent activities of a business process. Furthermore, they can be used to capture routing conditions of a process. So-called places define the pre-conditions and post-conditions of these transitions. Accordingly, places and transitions have to alternate in the Petri net graph, and directed arcs are used to connect them. The notation of Petri nets symbolizes transitions as rectangles and places as circles. The behavioral semantics of Petri nets relate to the concept of a marking (or state). A marking assigns none, one, or multiple so-called tokens (represented by a dot) to each place of a Petri net. A Petri net together with a marking is called system. The behavior of a system is defined by the firing rule: a marking of a system enables a transition if and only if all its predecessor places have at least one token assigned. In this case, a new marking is reached from the current marking by firing the enabled transition such that the new marking equals the current marking minus one token in each predecessor place of the enabled transition and plus one token in each successor place of the enabled transition.

Figure 3 shows a Petri net model for a loan request process inspired by Nüttgens and Rump (2002). The net belongs to the class of workflow nets, these are Petri nets with one start and one end place. As a first transition in the net, record loan request has to be executed. After conduct risk assessment, there is a decision to either continue with negative risk assessment or with positive risk assessment: firing the transition conduct risk assessment produces a token in the subsequent place which can be consumed by one of the following transitions. In case of a negative risk assessment, there is the option to repeat the risk assessment if check client assessment yields a positive result. If that is not the case, the reject loan request transition is executed. In case of a positive risk assessment, there is another choice depending on whether the client is a new client or not. The dot in the subsequent place indicates that the state of the system now permits to make this choice. If the client is new, the right column of transitions is executed in addition to setting up the loan contract. This behavior is provided by the requester is new client transition producing a token on both its output places which introduces parallelism. The transition without a label synchronizes the two transitions sign loan contract and offer further products which are meant to be executed for any client. This transition produces a token in the end place signalling the completion of the process.

Petri nets are often used not only for the modeling of business processes and workflows, but also for the verification of desirable behavior. The so-called soundness property (van der Aalst, 1997) defined for workflow nets is of pivotal importance in this context. It demands that a process should have (1) the option to complete, (2) that proper completion is guaranteed, and (3) that there are no dead tasks that will never be executed. Based on a Petri net and an initial marking, one can check soundness by calculating the so-called reachability graph. The reachability graph represents all states that can be reached in a Petri net as nodes and all permitted state changes as directed arcs between these nodes. If the net is large and there is a high degree of parallelism, such a verification approach might suffer from the state explosion problem (Valmari, 1998). This problem can be partially solved by applying reduction rules (see Berthelot, 1987, 1986; Esparza, 1994). Deadlocks are a prominent type of error that results in a net to be unsound. In the example process of Figure 3, there is a deadlock if the client is not a new client. In this case the requester is not a new client transition consumes the token of the current marking and forwards it to its successor place. Eventually, the loan contract is set up transition produces two tokens. The right

Figure 3. Petri net for a loan request process with a deadlock



one which is an input to the offer further products transition can actually never be forwarded, since the transition will never receive a another token at its second input place. Therefore, this marking is called a deadlock, since no state change is possible even though the final state has not yet been reached. In practice, such verification analysis has to be performed by tools. An example of an open source tool for the verification of soundness is Woflan (Verbeek et al., 2001). For an extensive introduction to Petri nets, the reader is referred to Desel and Esparza (1995), Murata (1989), and Reisig and Rozenberg (1998).

EPCs

The event-driven process chain (EPC) is a business process modeling language for the representation of temporal and logical dependencies of activities in a business process (Keller, Nüttgens, & Scheer, 1992). It is utilized in the architecture of integrated information systems (ARIS) by Scheer (1998, 2000) as the central method for the conceptual integration of the functional, organizational, data, and output perspective in information systems design. The EPC syntax offers function type elements to capture activities of a process and event type elements describing pre-conditions and post-conditions of functions. Furthermore, there are three kinds of connector types including AND (symbol $\wedge$), OR (symbol $\vee$), and XOR (symbol $\times$) for the definition of complex routing rules. Connectors have either multiple incoming and one outgoing arc (join connectors) or one incoming and multiple outgoing arcs (split connectors). As a syntax rule, functions and events have to alternate, either directly or indirectly when they are linked via one or more connectors. Furthermore, OR- and XOR-splits after events are not allowed, since events cannot make decisions. Control flow arcs are used to link these elements. The EPC notation represents functions as rounded rectangles, events as hexagons, and connectors as circles with the respective symbol in it.

The behavioral semantics of an EPC can be described as follows. The AND-split activates all subsequent branches in concurrency. The XOR-split represents a choice between one of several alternative branches. The OR-split triggers one, two or up to all of multiple branches based on conditions. In both cases of the XOR- and OR-split, the activation conditions are given in events subsequent to the connector. Accordingly, splits from an event to multiple functions are forbidden with XOR and OR as the activation conditions do not become clear in the model. The AND-join waits for all incoming branches to complete, and then it propagates control to the subsequent EPC element. The XOR-join merges alternative branches. The OR-join synchronizes all active incoming branches. This feature is called non-locality since the state of all transitive predecessor nodes has to be considered. There are several approaches towards the formalization of EPC semantics (see Kindler, 2006; Mendling & van der Aalst, 2007). The most recent formalization is defined by Mendling (2007).

Figure 4 shows an EPC model for the same loan request process as described in Nüttgens and Rump. The start event loan is requested and signals the start of the process and the precondition to execute the record loan request function. After the post-condition request is recorded, the process continues with the function conduct risk assessment after the XOR-join connector. The subsequent XOR-split connector indicates a decision. In case of a negative risk assessment, the function check client assessment is performed. The following second XOR-split marks another decision: in case of a negative client assessment the process ends with a rejection of the loan request; in case of a positive client assessment, the conduct risk assessment function is executed a second time under consideration of the positive client assessment. If the risk assessment is not negative, there is another decision point to distinguish new clients and existing clients. In case of an existing client, the set up loan contract function is conducted. After that, the AND-split indicates that two activities have to be executed: first, the sign loan contract function; and second, the offer further products function. If the client is new, the analyze requirements function has to be performed in addition to setting up the loan contract. The OR-join waits for both functions to be completed if necessary. Therefore, there is no deadlock here. If the analyze requirements function will not be executed in the process, it continues with offer further products immediately.

EPCs are often used for modeling business processes on a conceptual level. Originally, EPCs were introduced to offer an understandable representation of processes, and experiments confirmed that users indeed seem to understand them more easily compared to Petri nets (Sarshar & Loos, 2005). On the other hand, the introduction of the OR-join and the option to define multiple start and end events has been a considerable challenge for the verification of EPCs. Dehnert and Rittgen (2001) argue that business processes are often conceptually modelled in such a way that only the desired behavior results in a proper completion. Since such models are not used for workflow execution, non-normative behavior is resolved by the people working in the process in a cooperative and ad-hoc fashion. Accordingly, they define a process to be relaxed sound if every transition in a Petri net representation of the EPC model is included in at least one proper execution sequence. This Petri net representation of the EPC can be derived if OR-joins are mapped to a Petri net block (see Dehnert, 2002). In this case, the Petri net state space is larger than the actual state space with synchronization. Furthermore, based on the relaxed soundness criterion, it is possible to check whether a join should synchronize (Dehnert & van der Aalst, 2004). The verification of relaxed soundness has, for instance, been implemented in the open source tool ProM (see van Dongen, van der Aalst, & Verbeek, 2005) and been used for the verification of the SAP reference model (van Dongen & Jansen-Vullers, 2005; Mendling et al., 2006b). Only recently, the property of EPC soundness has been defined by Mendling and van der Aalst (2006, 2007), based on the EPC semantics formalization in Mendling (2007).

YAWL

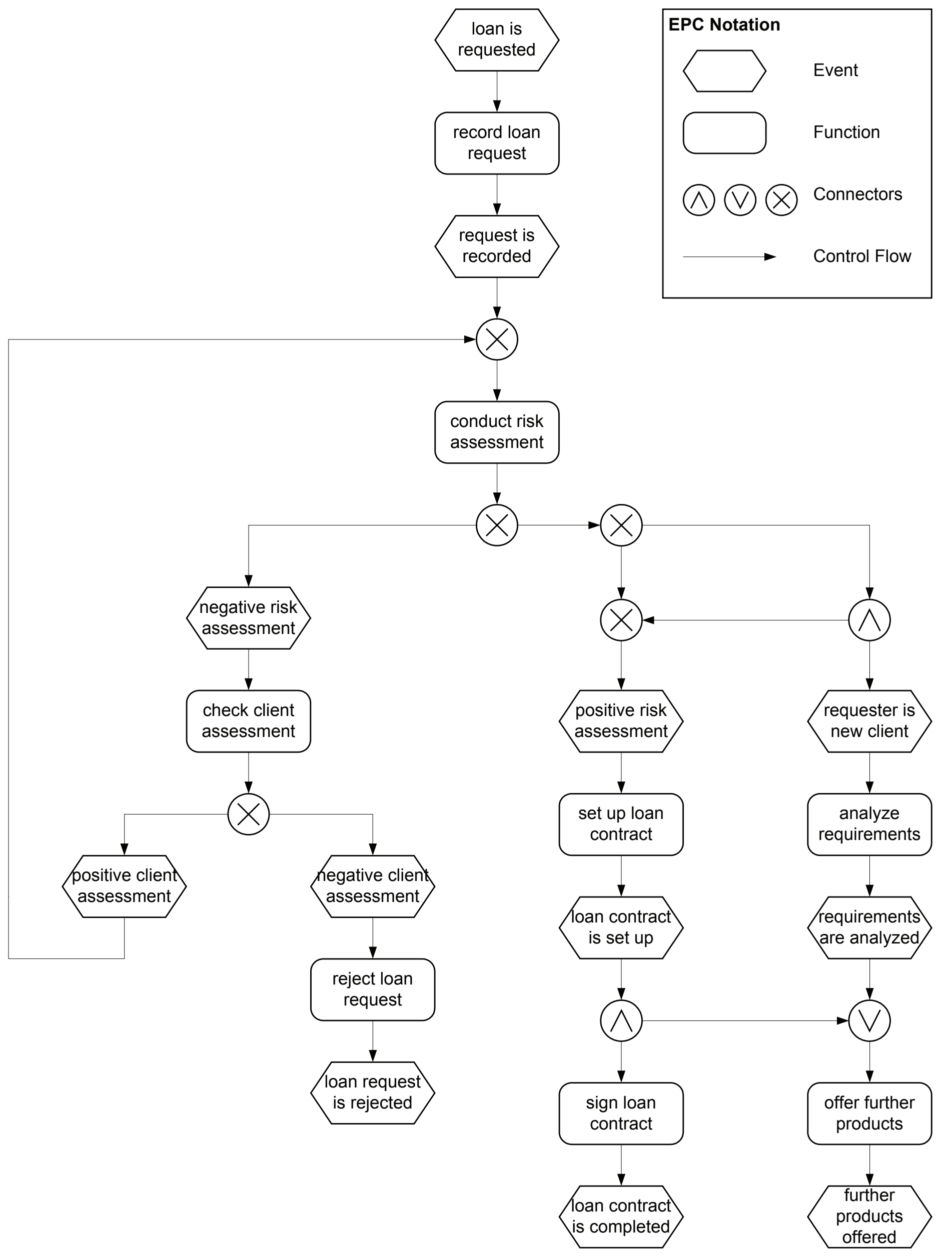
Triggered by their analysis of control flow modeling support by existing workflow systems, van der Aalst et al. (2003) identify a set of 20 so-called workflow patterns. These patterns cover different behavioral properties of a process that one might want to express by the help of a modeling language. The analysis of both Petri nets and EPCs revealed that these languages have problems to represent certain behavior. While EPCs are not able to express state-based patterns properly, Petri nets do not support advanced synchronization semantics such as defined by the OR-join. Furthermore, both languages do not provide a mechanism to express multiple instantiation and cancellation. YAWL (yet another workflow language) has been defined to directly support the specification of all patterns (van der Aalst & ter Hofstede, 2005).[8] As a consequence, YAWL can be considered a superset of both Petri nets and EPCs in terms of modeling elements.

The YAWL syntax includes conditions that match Petri net places and tasks that match Petri net transitions. Similar to Petri nets, conditions and tasks have to alternate, but single conditions between two tasks can be omitted. YAWL tasks optionally include a split and a join condition of type AND, XOR, or OR with basically the same semantics as the respective EPC connectors. Beyond that, a multiple instance task might be executed several times in parallel until a condition is fulfilled. Such multiple instance tasks specify four parameters: the minimum and the maximum number of instances, a threshold for continuation, and whether new instances may be created dynamically or not. Furthermore, after a task is completed a specified set of other tasks in a cancellation area may be cancelled. Using this mechanism, it is easy to expression exceptional behavioral when a process must be terminated abnormally. The YAWL notation is illustrated on the right-hand top of Figure 5.

Figure 5 also shows the loan request process as a YAWL model. The process is modelled in a similar way as the Petri net model. Still, the OR-join before offer further products resolves the deadlock problem. The condition after the first task illustrates that conditions may be explicitly modelled, but YAWL also permits to leave them out. Furthermore, this task is defined as a multiple instance task. Its parameters indicate that it has to be executed at least zero times, but at maximum three times. Offering only three products to a customer prevents the customer to become tried of offers. There is no threshold, but it is possible to create new instances dynamically, that is, if the clerk identifies demand for a further product, she can easily start offering it.

Based on the YAWL workflow language, a whole open source workflow system has been built which is called YAWL system (van der Aalst et al., 2004). There is several verification features included in the YAWL modeller (see Wynn, Edmond, van der

Figure 4. EPC for a loan request process (Nüttgens & Rump, 2002)



Aalst, & ter Hofstede, 2005; Wynn, Verbeek, van der Aalst, ter Hofstede, & Edmond, 2006).

UML Activity Diagrams

Activity diagrams belong to the diagram family of the unified modeling language (UML) specified by the object management group (OMG, 2004). Since UML, and also activity diagrams, cover an extensive set of notation elements, we focus only on those ones which are used in the workflow pattern analysis reported by Wohed, van der Aalst, Dumas, ter Hofstede, and Russell (2005). The basic syntax elements of UML activity diagrams are actions and control nodes. An action transforms a set of inputs into a set of outputs which implies a state change of the system. Wohed et al. (2005), in particular, focus on the action subtypes send signal action and accept event action in their discussion of control flow modeling with UML activity diagrams. The send signal action creates a signal which may be received by a matching accept event action. The initial node and the activity final representing the start and the completion of a process belong to the set of control nodes. Furthermore, a decision branches to one of alternative branches based on conditions. These alternative branches flow together at merge nodes. The folk can be used to introduce parallelism which can be synchronized with a join. The semantics of these routing elements are similar to that of the respective EPC and YAWL splits and joins. The notation of these different elements is displayed in Figure 6. There is an extensive tool support for modeling UML activity diagrams in practice. Furthermore, there is some research on adopting Petri net verification techniques for UML activity diagrams reported in Eshuis (2002) and Eshuis and Wieringa (2003).

BPMN

The business process modeling notation (BPMN) started as an working group within the business process management initiative (BPMI). Since the merge of BPMI and OMG, BPMN has also become an OMG standard (see OMG, 2006). Figure 7 shows the notation of the main BPMN syntax elements. BPMN covers tasks and events. But in contrast to EPCs, these elements do not have to alternate. The semantics of so-called gateways is similar to the respective joins and splits of EPCs and YAWL. For a complete introduction to BPMN, refer to the specification (OMG, 2006). In practice, there is a growing BPMN support by tool vendors. There are also some academic works on the verification of BPMN models (Puhlmann & Weske, 2006).

Comparison Based on Routing Elements

In this section, we consider the six different connectors of EPCs, that is, XOR-split and XOR-join, AND-split and AND-join, OR-split and OR-join, for the comparison of the different modeling languages. Table 1 takes these routing elements as a benchmark, and shows that the behavioral semantics of XOR-connectors and AND-connectors, as well as OR-split connectors, can be represented in all the considered languages. In workflow nets XOR-connectors and AND-connectors are captured by places and transitions with multiple input and output arcs, respectively. OR-split behavior can be specified as a complex sub-net that determines each possible combination of inputs. OR-join behavior cannot be modelled directly, but a relaxed soundness analysis is possible. In UML activity diagrams the XOR-split maps to a decision, the XOR-join to a merge, the AND-split to a fork, the AND-join to a join, and the OR-split to a fork with guards on its output arcs. OR-joins cannot be represented in UML activity diagrams directly. In BPMN, routing elements are called gateways. Basically, each EPC connector can be transformed to a respective gateway. In YAWL, there are also similar splits and joins matching the behavior of the EPC connectors.

Comparison Based on Workflow Patterns

The workflow patterns identified by van der Aalst et al. (2003) can be utilized to clarify semantics or to serve as a benchmark. Table 2 illustrates the result of several workflow pattern analyses of EPCs (Mendling et al., 2005b), Workflow nets (van der Aalst, 1997), UML activity diagrams (Wohed et al., 2005), BPMN (Wohed et al., 2006), and YAWL (van der Aalst & ter Hofstede, 2005). It can be seen that EPCs support the basic control flow patterns, multiple choice, and synchronizing merge. These patterns can be directly represented with the different EPC connectors. Furthermore, EPCs permit

Figure 5. YAWL model for the loan request process

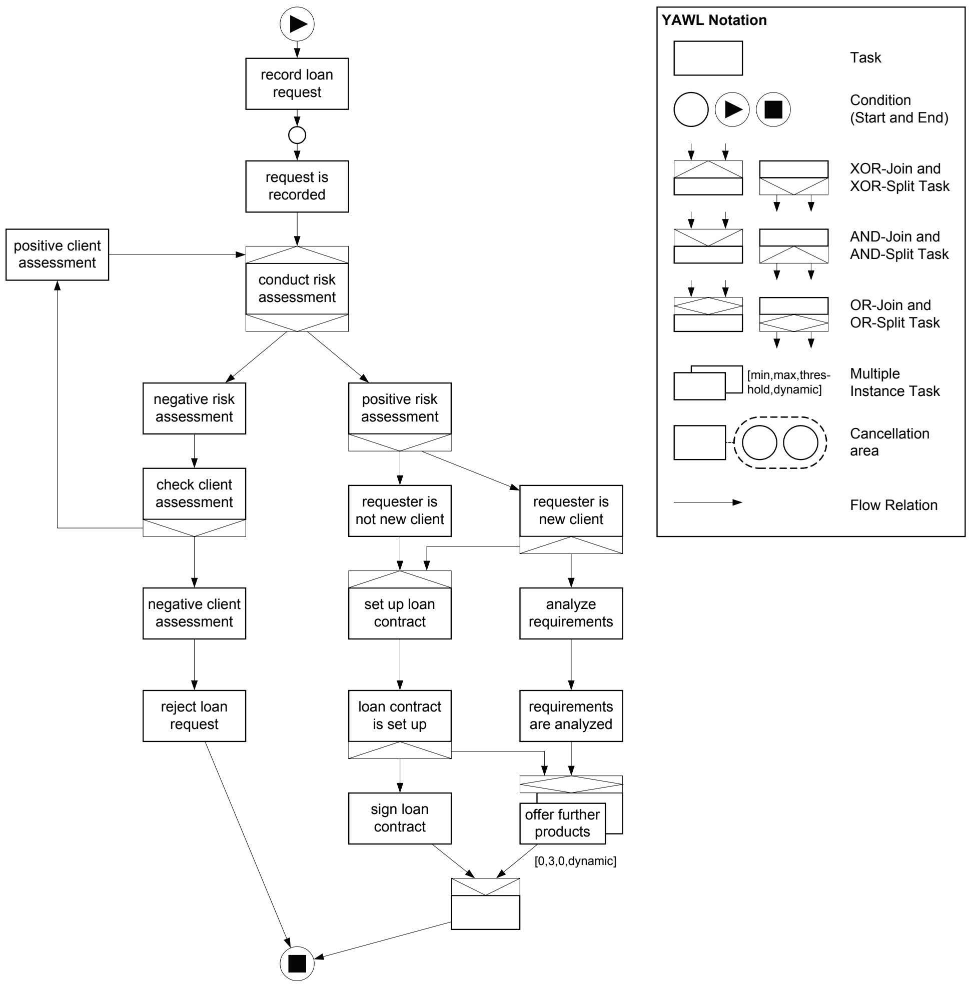


Figure 6. UML notation of important control flow elements

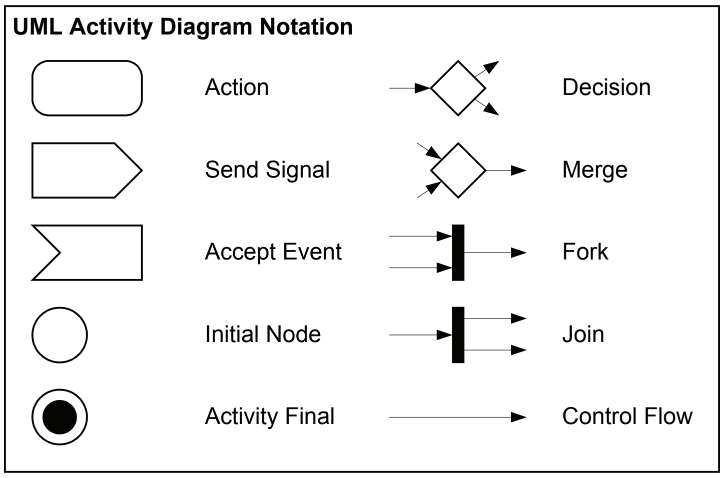


Figure 7. BPMN notation of important control flow elements

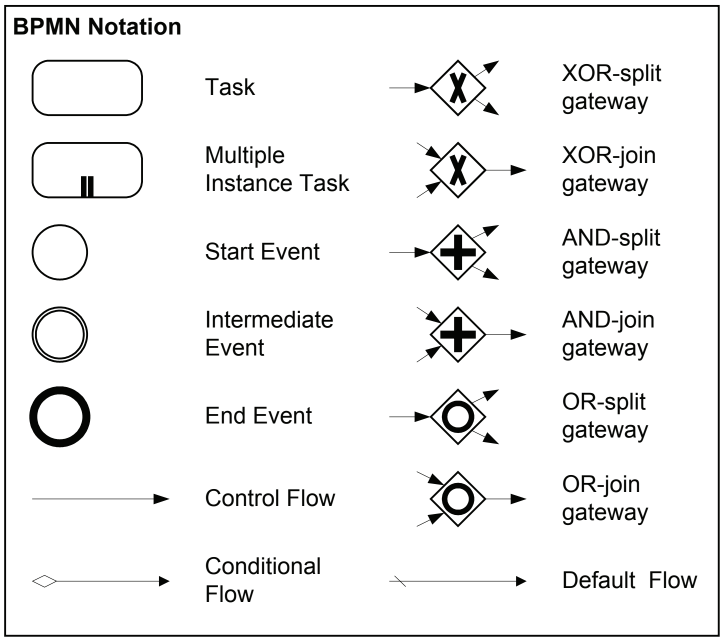


Table 1. EPC routing elements and equivalent elements in other business process modeling languages

EPC	workflow nets	UML AD	BPMN	YAWL
XOR-split	multi-out place	Decision	XOR-gateway	XOR-split task
XOR-join	multi-in place	Merge	XOR-gateway	XOR-join task
AND-split	multi-out transition	Fork	AND-gateway	AND-split task
AND-join	multi-in transition	Join	AND-gateway	AND-join task
OR-split	complex subnet	Fork	OR-gateway	OR-split task
OR-join	-	-	OR-gateway	OR-join task

arbitrary cycles and offer implicit termination. Multiple instances with a priori design time knowledge can be modelled by an AND-block, with as many instances as required of the same activity in parallel. The yEPC extension of EPCs provides support for all patterns (see Mendling et al., 2005c).

In contrast to EPCs, workflow nets support the state-based patterns, but perform weak when it comes to advanced branching and synchronization patterns. UML activity diagrams cover several patterns missing only the synchronizing merge, multiple instances without apriori runtime knowledge, and two state-based patterns. BPMN performs even better since it supports the synchronizing merge, but only in a structured block. As YAWL was defined to provide a straight-forward support for the workflow patterns, it is no surprise that it has the best score. The implicit termination pattern is not supported in order to force the designer to make the completion condition explicit. The comparison reveals that BPMN and YAWL support most of the patterns. Yet, they offer much more syntax elements than EPCs and Petri nets, which might imply that users would need more time to learn them. In practice, the choice for or against a language should be taken with care, considering also the tool support for modeling and verification. For more details on the workflow patterns, refer to van der Aalst et al. (2003).

Modeling Methods

It is a fundamental insight of software engineering that design errors should be detected as early as possible (see Boehm, 1981; Moody, 2005; Wand & Weber, 2002). The later errors are detected, the more rework has to be done, and the more design effort has been at least partially useless. This also holds for the consecutive steps of analysis, design, and implementation in the business process management life cycle (see Philippi & Hill, 2007; Rosemann, 2006). In the design phase, process models are typically created with semi-formal business process modeling languages while formal executable models are needed for the implementation. This issue is often referred to as the gap between business process design and implementation phase (see zur Muehlen & Rosemann, 2004). Therefore, the guidelines of process modeling stress correctness as the most important quality attribute of business process models (Becker et al., 2000).

In order to provide a better understanding of potential modeling errors, we consider the information modeling process as identified by Frederiks and Weide (2006) as a modeling method. This process can also serve as a framework for discussing business process modeling in the analysis and design phase of the business process management life cycle. Furthermore, it covers several steps to provide quality assurance in the modeling phase, which is of paramount importance for the success of modeling projects (see Rosemann, 2006). Figure 8 gives a business process modeling process mainly inspired by Frederiks and Weide (2006) and consisting of eight steps. In accordance with van Hee, Sidorova, Somers, and Voorhoeve (2006), we propose to first verify the process model (step 6) before validating it (step 7-8).

The business process modeling process starts with collecting information objects relevant to the domain (step 1). Such information objects include documents, diagrams, pictures, and interview recordings. In step 2, these different inputs are verbalized to text that serves as a unifying format. This text is rearranged according to some general guideline of how to express facts (step 3) yielding an informal specification. The following step (step 4) takes this informal specification as a basis to discover modeling concepts from and to produce a normalized specification. This normal form specification is then mapped to constructs of the process modeling language (step 5) in order to create a business process model. These models have to be verified for internal correctness (step 6) before they can be paraphrased back to natural language (step 7) in order to validate them against the specification (step 8). In steps 6-8, the order of activities follows the proposal by van Hee et al. (2006). It is a good idea to first verify the internal correctness of a model before validating it against the specification, as this prevents incorrect models from being unnecessarily validated.

The business process modeling process points to two categories of potential errors based on the distinction between verification and validation. This distinction follows the terminology of different authors, including Boehm (1979), Hoppenbrouwers, Proper, and van der Weide (2005), Sommerville (2001), and Valmari (1998). Different terms for similar concepts are used by Soffer and Wand (2004).

- Verification addresses both the general properties of a model and the satisfaction

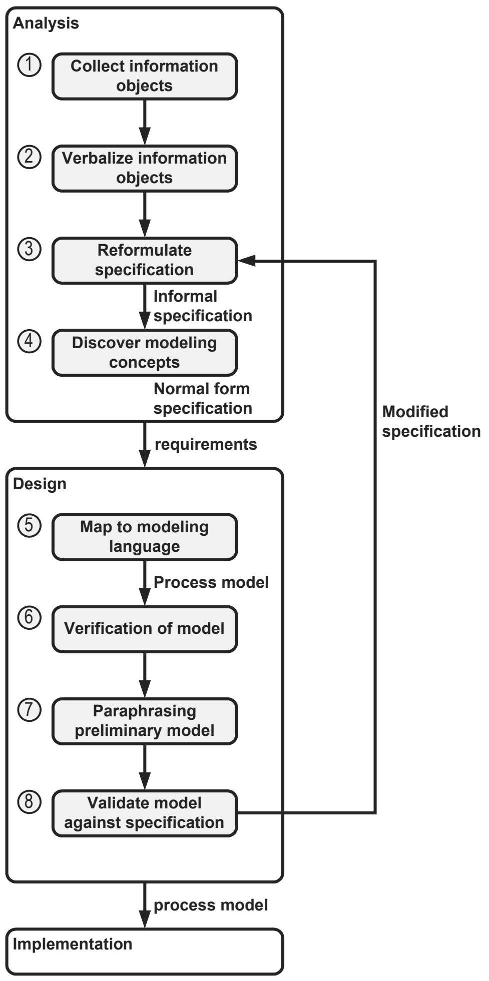
Table 2. Workflow pattern support of EPCs and other business process modeling languages. + means pattern supported, +/- partially supported, and - not supported

Workflow Pattern	EPC	Wf. nets	UML AD	BPMN	YAWL
Basic Control Flow Patterns					
1. Sequence	+	+	+	+	+
2. Parallel Split	+	+	+	+	+
3. Synchronization	+	+	+	+	+
4. Exclusive Choice	+	+	+	+	+
5. Simple Merge	+	+	+	+	+
Advanced Branching and Synchronization Patterns					
6. Multiple Choice	+	+	+	+	+
7. Synchronizing Merge	+	-	-	+/-	+
8. Multi Merge	-	+	+	+	+
9. Discriminator	-	-	+	+/-	+
Structural Patterns					
10. Arbitrary Cycles	+	+	+	+	+
11. Implicit Termination	+	-	+	+	-
Patterns involving Multiple Instantiation (MI)					
12. MI without Synchronization	-	+	+	+	+
13. MI with apriori Design Time Knowledge	+	+	+	+	+
14. MI with apriori Runtime Knowledge	-	-	+	+	+
15. MI without apriori Runtime Knowledge	-	-	-	-	+
State-based Patterns					
16. Deferred Choice	-	+	+	+	+
17. Interl. Parallel Routing	-	+	-	+/-	+
18. Milestone	-	+	-	-	+
Cancellation Patterns					
19. Cancel Activity	-	+/-	+	+	+
20. Cancel Case	-	-	+	+	+

of a given formula by a model. Related to the first aspect, formal correctness criteria play an important role in process modeling. Several criteria have been proposed including soundness for Workflow nets (van der Aalst, 1997), relaxed soundness (Dehnert & Rittgen, 2001), or well-structuredness (see Dehnert & Zimmermann, 2005). The second aspect is the subject of model checking and involves issues like separation of duty constraints, which can be verified, for example, by using linear temporal logic (see Pnueli, 1977).

- Beyond that, validation addresses the consistency of the model with the universe of discourse. As it is an external correctness criterion, it is more difficult and more ambiguous to decide. While verification typically relies on an algorithmic analysis of the process model, validation requires the consultation of the specification and discussion with business process stakeholders. SEQUAL can be used as a conceptual framework to validate different quality aspects of a model (Krogstie et al., 2006, Lindland et al., 1994).

Figure 8. Business process modeling process in detail, adapted from Frederiks and Weide (2006)


Analysis
1 Collect information objects
2 Verbalize information objects
3 Reformulate specification
Informal specification
4 Discover modeling concepts
Normal form specification
requirements
Modified specification
Design
5 Map to modeling language
Process model
6 Verification of model
7 Paraphrasing preliminary model
8 Validate model against specification
process model
Implementation

In practice, a well-structured modeling process is not always followed. Recent empirical research by Rittgen (2007) and Stirna, Persson, and Sandkuhl (2007) shows that social aspects and organizational constraints have a strong influence on how the modeling processes is actually conducted. Beyond that, verification and validation depends on respective support provided by tools for the language of choice. Several surveys stress the importance of quality assurance in process modeling. The verification of the EPC models of the SAP reference model has shown that there are several formal errors in the models (Dongen & Jansen-Vullers, 2005; Dongen et al., 2005, Mendling et al., 2006a; Zukunft & Rump, 1996). In Mendling et al. (2006a), the authors identify a lower bound for the number of errors of 34 (5.6%), using the relaxed soundness criterion. Based on EPC soundness, Mendling (2007) finds 126 models with errors (20.9%). In another survey, Gruhn and Laue (2007) analyze a collection of 285 EPCs mainly taken from master theses and scientific publications. From these 285 models, 30% had trivial errors and another 7% had non-trivial errors. Up to now, tool support for verification and validation of business process models is rather weak which might partially explain the high error rates. This makes it even more important to follow a structured modeling process to detect and correct errors as early as possible.

FUTURE TRENDS

The standardization and the seamless transformation from conceptual business process models to executable workflow models is a current trend in research and industry. BPMN was created with the ambition to provide a direct mapping to executable BPEL web service processes. BPEL, that is, business process execution language for Web service, is at the time of writing in the final phase of standardization of its second version. This section presents standardization efforts with a focus on the concepts of BPEL and its support by major software vendors.

Finally, we discuss some transformation issues that are only partially solved up to now.

Business Process Execution and Standardization

The standardization of business process management and workflow technology has been discussed for more than 10 years (see Hollingsworth, 2004). Several standardization bodies have proposed specifications for different aspects of business process management. The five bodies that have gained the most attention in this context are WfMC, OMG, BPMI, W3C, and OASIS. The WfMC (workflow management coalition) has been the first organization to promote workflow standards. Its workflow reference model distinguishes five interfaces of a workflow system (Hollingsworth, 1994). From WfMC's set of specifications,[9] the XPDL standard for process definition (interface 1) is the most prominent one (see Workflow Management Coalition, 2005). The BPMI (business process management initiative) started off in 2000 as an industry consortium to promote business process management standards. In 2002, BPMI published BPML (Arkin, 2002), an XML-based language for the specification of executable processes with Web service interaction, and in 2004, the standardization of BPMN as a visual notation started. The OMG (Object Management Group) first got involved with workflow technology as they accepted the workflow management facility specification as a standard in 1998. In 2005, OMG and BPMI agreed to merge their business process related activities. As a consequence, BPMN is now an OMG standard (see OMG, 2006). The W3C (World Wide Web Consortium) has published several standards for web service choreography. Choreography describes the interaction of distributed processes from a global point of view. WS-CDL (Kavantzas et al., 2005) is the most recent specification in this area. It is meant to be utilized in conjunction with process definition languages that define the private implementation of processes. OASIS (Organization for the Advancement of Structured Information Standards) is an industry group that defines XML-based standards for Web services and business integration. OASIS participates, for example, in the specification of the ebXML framework. For further details on business process related standards see (Mendling et al., 2005b).

Since 2003, OASIS is also responsible for the standardization of BPEL.[10] The work on BPEL started with a merger of IBM's WSFL process definition specification with Microsoft's XLANG which resulted in the first version of BPEL (Curbera et al., 2002). In 2003 BEA, SAP, and Siebel joined in to extend BPEL to version 1.1 (Andrews et al., 2003). Currently, the second version of BPEL is in the final phase of standardization (see Alves et al., 2007).[11]

Main Concepts of BPEL

BPEL is an XML-based language that models a business process as a composition of elementary Web services. A so-called BPEL engine is a dedicated software component that is able to execute BPEL process definitions. Each BPEL process can be accessed as a Web service of the BPEL engine, too. The BPEL specification depends on the W3C standards WSDL for Web service description, XML schema for the definition of data structures, and XPath for retrieval of XML elements. Six of BPEL's most important concepts are briefly presented in the following, that is, partner links, variables, correlation, basic activities, structured activities, and handlers. We will use the element names of BPEL 2.

- **Partner links:** A partner link provides a communication channel to remote Web services which are utilized in the BPEL process. A respective partner link type must be defined first to specify the required and provided WSDL port types.
- **Variables:** Variables are used to store both message data of Web service interactions and

control data of the process. A variable must be declared in the header of a BPEL process by referencing a WSDL or an XML Schema data type.

- **Correlation:** As BPEL supports long-running business processes, there may be several process instances waiting for Web service messages at a certain point of time. A correlation set specifies so-called properties, that is, XPath statements to retrieve message parts that are unique for a specific process instance. According to a certain property value, like, for example, ordernumber=1002006, a message is handed to the matching process instance.
- **Basic activities:** The basic steps of a BPEL process are performed by basic activities. There are activities to send and receive messages from Web services (receive, invoke, reply), to change the content of variables (assign), to wait for a certain period or up to a certain point in time (wait), or to terminate the process (exit, formally called terminate).[12] The second version of BPEL introduces an activity to check conformance to a schema (validate) and the possibility to add proprietary activities (extensionActivity).
- **Structured activities:** The control flow of basic activities can be defined in two different styles: block-oriented or graph-based. Both styles can be mixed. Block-oriented control flow can be defined with structured activities. BPEL offers activities to specify parallel execution (flow), conditional branching based on data (if-else) or on receipt of a message (pick), sequential execution (sequence), and different loops (while, repeatUntil, forEach). Structured activities can be nested. Scopes are special structured activities. They mark-off the scope of local variables and handlers. Control flow can also be defined graph-based, but without introducing cycles, using so-called links. A link represents synchronization between two activities.
- **Handlers:** BPEL provides handlers to deal with unexpected or exceptional situations. Event handlers wait for messages or time events. They can be used to specify deadlines on the process level. Fault handlers catch internal faults of the BPEL process. If the fault cannot be cured, the compensation handler can be triggered to undo the effects of already completed activities. Finally, the termination handler offers a mechanism to force a process to terminate, for example, due to external faults.

Even though BPEL supports a rich set of primitives to specify executable processes, there are still some features missing towards full-fledged business process specification. The extension activity of BPEL 2 is a useful anchor point to fill these gaps. Currently, there are several BPEL extensions in progress of development, in particular BPELJ[13] for Java inline code, BPEL4People[14] for human worklists, and BPEL-SPE[15] for sub-processes.

BPEL Support

Several major software vendors support the standardization of BPEL.[16] Several of these companies already provide BPEL support in their products. Furthermore, there are also open source implementations of BPEL including ActiveBPEL, bexee, MidOffice, and Twister. Therefore, it should be expected that BPEL will soon become not only a de-iure, but a de-facto standard for the definition of executable business processes.

For vendors, there are basically two options to align with BPEL, either to develop a BPEL engine and related tools from scratch or to implement BPEL import and export for existing systems. Oracle is one of the few vendors who offer a generic implementation of BPEL called Oracle BPEL Process Manager. Several other companies have chosen to extend their systems with import and export. In this case, the BPEL process has to be mapped to the object structure of the target system. The mapping between BPEL and graph-based control flow, in particular with unstructured loops, is an interesting challenge for both academia and practice. So-called transformation strategies (Mendling et al., 2006) can serve as a blue print for that. Van der Aalst and Lassen (2006) define a transformation from Petri nets to BPEL using reduction rules. Ouyang, van der Aalst, Dumas, and ter Hofstede (2006) propose a transformation of unstructured loops to event-condition-action rules that are implemented via BPEL event handlers. Another option, but not in the general case, is to derive structured from unstructured loops (Zhao, Hauser, Bhattacharya, Bryant, & Cao, 2006). The simplest solution to this problem for the vendors is to prohibit the definition of unstructured cycles in their process design tool. Then all import and export transformations between BPEL and internal graph-based representation can be implemented without loss of information.

CONCLUSION

This chapter provided an overview of business process management and business process modeling. In particular, we elaborated on the background of business process management by giving a historical classification of seminal work, and by discussing the business process management life cycle. In this life cycle, business process models play an important role. We approached business process modeling from a general information systems point of view and deducted a definition. Furthermore, we presented several frequently-used business process modeling languages including Petri nets, EPCs, YAWL, UML activity diagrams, and BPMN, as well as modeling methods for business process modeling. As a future trend, we identified standardization and executability of business process models. In particular, we focused on recent standardization efforts and research challenges related to the business process execution language for Web services. We expect this area to be the driver for new concepts and innovations within the next couple of years.

REFERENCES

Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., et al. (2007). Web services business process execution language version version 2.0. Committee specification 31 January 2007, OASIS.

Ami T., & Sommer, R. (2007). Comparison and evaluation of business process modelling and management tools. *International Journal of Services and Standards*, 3(2), 249-261.

Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., et al. (2003). Business process execution language for Web services, Version 1.1. Specification, BEA Systems, IBM Corp., Microsoft Corp., SAP AG, Siebel Systems.

Arkin, A. (2002). *Business process modeling language (BPML)*. Spec., BPMI.

Atkinson, C., & Kühne, T. (2001a). Processes and products in a multi-level metamodeling architecture. *International Journal of Software Engineering and Knowledge Engineering, 11*(6), 761-783.

Atkinson, C., & Kühne, T. (2001b). The essence of multilevel metamodeling. In M. Gogolla & C. Kobryn (Eds.), *UML 2001—The unified modeling language, modeling languages, concepts, and tools, Proceedings of the 4th International Conference, Toronto, Canada* (Vol. 2185, pp. 19-33). Springer.

Atkinson, C., & Kühne, T. (2003). Model-driven development: A metamodeling foundation. *IEEE Software, 20*(5), 36-41.

Austin, J. L. (1962). *How to do things with words*. Cambridge, MA: Harvard University Press.

Batini, C., Lenzerini, M., & Navathe, S. B. (1986). A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys, 18*(4), 323-364.

Becker, J., & Kugeler, M. (2003). *Process management: A guide for the design of business processes*. Springer-Verlag.

Becker, J., Rosemann, M., & Schütte, R. (1995). Grundsätze ordnungsmässiger Modellierung. *Wirtschaftsinformatik, 37*(5), 435-445.

Becker, J., Rosemann, M., & von Uthmann, C. (2000). Guidelines of business process modeling. In W. M. P. van der Aalst, J. Desel, & A. Oberweis (Eds.), *Business process management. Models, techniques, and empirical studies* (pp. 30-49). Berlin: Springer.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American*, *May.*

Berthelot (1986). Checking properties of nets using transformations. In G. Rozenberg (Ed.), *Advances in Petri nets 1985* (Vol. 222, pp. 19-40). Berlin: Springer-Verlag.

Berthelot, G. (1987). Transformations and decompositions of nets. In W. Brauer, W. Reisig, & G. Rozenberg (Eds.), *Advances in Petri nets 1986 Part I: Petri Nets, central models and their properties* (Vol. 254, pp. 360-376). Berlin: Springer-Verlag.

Boehm, B. W. (1979). *Research directions in software technology.* MIT Press.

Boehm, B. W. (1981). *Software engineering economics*. Englewood Cliffs: Prentice-Hall.

Bunge, M. (1977). *Treatise on basic philosophy* (vol.3). *Ontology I. The furniture of the world*. New York: D. Reidel Publishing.

Casati, F., Ceri, S., Pernici, B., & Pozzi, G. (1995). Conceptual modeling of workflows. In *Proceedings of the OOER International Conference*, Gold Cost, Australia.

Chen, P. (1976). The entity-relationship model—towards a unified view of data. *ACM Transactions on Database Systems (TODS), 1*, 9-36.

Curbera, F., Goland, Y., Klein, J., Leymann, F., Roller, D., Thatte, S., et al. (2002). *Business process execution language for Web services, version 1.0. specification, BEA systems*. IBM Corp., Microsoft Corp.

Davenport, T. H. (1993). *Process innovation: Reengineering work through information technology.* Boston: Harvard Business School Press.

Davies, I., Green, P., Rosemann, M., Indulska, M., & Gallo, S. (2006). How do practitioners use conceptual modeling in practice? *Data & Knowledge Engineering, 58*(3), 358-380.

Dayal, U., Hsu, M., & Ladin, R. (2001, September). Business process coordination: State of the art, trends, and open issues. In P. M. G. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, & R. T. Snodgrass (Eds.), *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB),* Roma, Italy.

Dehnert, J. (2002). Making EPC´s fit for workflow management. In M. Nüttgens & F. J. Rump (Eds.), *Proceedings of the 1st GI-Workshop on Business Process Management with Event-Driven Process Chains EPK 2002,* Trier, Germany, (pp. 51-69).

Dehnert, J., & van der Aalst, W. M. P. (2004). Bridging the gap between business models and workflow specifications. *International J. Cooperative Inf. Syst., 13*(3), 289-332.

Dehnert, J., & Rittgen, P. (2001). Relaxed soundness of business processes. In K. R. Dittrick, A. Geppert, & M. C. Norrie (Eds.), *Proceedings of the 13th International Conference on Advanced Information Systems Engineering* Vol. 2068, pp. 151-170,). Interlaken: Springer.

Dehnert, J., & Zimmermann, A. (2005, September 5-8). On the suitability of correctness criteria for business process models. In W. M. P. van der Aalst, B. Benatallah, F. Casati, & F. Curbera (Eds.), *Proceedings of the Business Process Management, 3rd International Conference, BPM 2005,* Nancy, France (Vol. 3649, pp. 386–391).

Desel, J., & Esparza, J. (1995). *Free choice Petri nets*, *volume 40 of Cambridge tracts in theoretical computer science*. Cambridge, UK: Cambridge Univ. Press.

Dumas, M., ter Hofstede, A., & van der Aalst, W. M. P. (2005). *Process aware information systems: Bridging people and software through process technology*. Wiley Publishing.

Ellis, C.A. (1979). Information control nets: A mathematical model of office information flow. In *Proceedings of the Conference on Simulation, Measurement and Modeling of Computer Systems* (pp. 225-240). Boulder, CO: ACM Press.

Ellis, C. A., & Nutt, G. J. (1980). Office information systems and computer science. *ACM Computing Surveys, 12*(1), 27-60.

Ellis, C. A., & Nutt, G. J. (1993). Modelling and enactment of workflow systems. In M. Ajmone Marsan (Ed.), *Application and theory of Petri nets* (Vol. 691, pp. 1-16). Berlin: Springer-Verlag.

Eshuis, H. (2002). *Semantics and verification of UML activity diagrams for workflow modelling*. PhD thesis, University of Twente, Enschede, The Netherlands.

Eshuis, H., & Wieringa, R. (2003). Comparing Petri nets and activity diagram variants for workflow modelling—a quest for reactive Petri nets. In H. Ehrig, W. Reisig, G. Rozenberg, & H. Weber (Eds.), *Petri net technology for communication based systems* (Vol. 2472). Berlin: Springer-Verlag.

Esparza, J. (1994). Reduction and synthesis of live and bounded free choice Petri nets. *Information and Computation, 114*(1), 50-87.

Fayol, H. (1966). *Administration industrielle et générale. Prévoyance, Organisation, Commandement, Coordination, Control*. Dunod.

Flatscher, R. G. (1998). *Meta-modellierung in EIA/CDIF*. ADV-Verlag, Wien.

Ford, H. (1926). *Today and tomorrow*. Doubleday, Page and Company.

Frederiks, P. J. M., & van der Weide, T. P. (2006). Information modeling: The process and the required competencies of its participants. *Data & Knowledge Engineering, 58*(1), 4-20.

Georgakopoulos, D., Hornick, M., & Sheth, A. (1995). An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases, 3*, 119-153.

Grochla, E., & Szyperski, N. (1975). *Information systems and organizational structure*. Walter de Gruyter.

Gruhn, V., & Laue, R. (2007). What business process modelers can learn from programmers. *Science of Computer Programming, 65*(1), 4-13.

Guizzardi, G., Herre, H., & Wagner, G. (2002, Octoer 7-11). On the general ontological foundations of conceptual modeling. In S. Spaccapietra, S. T. March, & Y. Kambayashi (Eds.), *Proceedings of the Conceptual Modeling—ER 2002, 21st International Conference on Conceptual Modeling,* Tampere, Finland (Vol. 2503, pp. 65-78). Springer.

Hammer, M., Champy, J. (1993). *Reengineering the corporation: A manifesto for business revolution*. New York: Harpercollins.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly, 28*(1), 75-105.

Hirschheim, R., & Klein, H. K. (1989). Four paradigms of information systems development. *Commun. ACM, 32*(10), 1199-1216.

Hofreiter, B., Huemer, C., & Kim, J.-H. (2006). Choreography of ebXML business collaborations. *Information Systems and E-Business Management.*

Hollingsworth, D. (1994). The workflow reference model. TC00-1003 Issue 1.1, Workflow Management Coalition.

Hollingsworth, D. (2004). *The workflow handbook 2004*, chapter The Workflow Reference Model: 10 Years On, pages 295–312. Workflow Management Coalition.

Hoppenbrouwers, S., Proper, H. A., & van der Weide, T. P. (2005, October 24-28). A fundamental view on the process of conceptual modeling. In L. M. L. Delcambre, C. Kop, H. C. Mayr, J. Mylopoulos, & O. Pastor (Eds.), *Proceedings of the Conceptual Modeling—ER 2005, 24th International Conference on Conceptual Modeling,* Klagenfurt, Austria, (Vol. 3716, pp. 128–143). Springer.

Hsu, M., & Kleissner, C. (1996). Objectflow: Towards a process management infrastructure. *Distributed and Parallel Databases, 4*(2), 169-194.

Jablonski, S., & Bussler, C. (1996). *Workflow management: Modeling concepts, architecture, and implementation*. London: International Thomson Computer Press.

Junginger, S., Kühn, H., Strobl, R., & Karagiannis, D. (2000). Ein Geschäftsprozessmanagement-werkzeug der nächsten Generation—Adonis: Konzeption und Anwendungen. *Wirtschaftsinformatik, 42*(5), 392-401.

Karagiannis, D., & Kühn, H. (2002). Metamodelling platforms. Invited Paper. In K. Bauknecht, A. Min Tjoa, & G. Quirchmayer (Eds.), *Proceedings of the 3rd International Conference EC-Web 2002—Dexa 2002,* Aix-en-Provence, France, (Vol. 2455, pp. 182-196).

Kavantzas, N., Burdett, D., Ritzinger, G., Fletcher, T., Lafon, Y., & Barreto, C. (2005). Web services choreography description language Version 1.0. W3C Candidate Recommendation 9 November 2005, World Wide Web Consortium, April 2005.

Keller, G., Nüttgens, M., & Scheer, A.-W. (1992). *Semantische Prozessmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)".* Heft 89, Institut für Wirtschaftsinformatik, Saarbrücken, Germany.

Kelly, S., Lyytinen, K., & Rossi, M. (1996, May 20-24). Metaedit+: A fully configurable multi-user and multi-tool case and came environment. In P. Constantopoulos, J. Mylopoulos, & Y. Vassiliou (Eds.), *Proceeedings of the Advances Information System Engineering, 8th International Conference, CAiSE'96,* Heraklion, Crete, Greece, (Vol. 1080, pp. 1-21). Springer.

Kindler, E. (2006). On the semantics of EPCs: Resolving the vicious circle. *Data & Knowledge Engineering, 56*(1), 23-40.

Kosiol, E. (1961). Modellanalyse als Grundlage unternehmerischer Entscheidungen. *Zeitschrift für betriebswirtschaftlicher Forschung,* 318–334.

Krogstie, J., Sindre, G., & Jørgensen, H. D. (2006). Process models representing knowledge for action: a revised quality framework. *European Journal of Information Systems, 15*(1), 91-102.

Kühne, T. (2006). Matters of (meta-) modeling. *Software and Systems Modeling, 5*(4), 369-385.

Leymann, F., & Roller, D. (2000). *Production workflow—concepts and techniques.* Prentice Hall.

Lindland, O. I., Sindre, G., & Sølvberg, A. (1994). Understanding quality in conceptual modeling. *IEEE Software, 11*(2), 42-49.

McGuinness, D. L., & van Harmelen, F. (2004). *OWL Web ontology language overview.* W3c recommendation, World Wide Web Consortium.

Mendling, J. (2007). *Detection and prediction of errors in EPC business process models.* PhD thesis, Vienna University of Economics and Business Administration.

Mendling, J., & van der Aalst, W. M. P. (2006). Towards EPC semantics based on state and context. In M. Nüttgens, F. J. Rump, & J. Mendling (Eds.), *Proceedings of the 5th GI Workshop on Business Process Management with Event-Driven Process Chains (EPK 2006)*, Vienna, Austria, (pp. 25-48). German Informatics Society.

Mendling, J., & van der Aalst, W. M. P. (2007). Formalization and verification of EPCs with OR-joins based on state and context. In J. Krogstie, A.L. Opdahl, & G. Sindre (Eds.), *Proceedings of the 19th Conference on Advanced Information Systems*

Engineering CAiSE 2007, Trondheim, Norway, (Vol. 4495, pp. 439-453). Springer-Verlag.

Mendling, J., & Hafner, M. (2005). From inter-organizational workflows to process execution: Generating BPEL from WS-CDL. In R. Meersman, Z. Tari, & P. Herrero (Eds.), *Proceedings of OTM 2005 Workshops*, (Vol. 3762).

Mendling, J., Lassen, K. B., & Zdun, U. (2006). Experiences in enhancing existing BPM tools with BPEL import and export. In J. L. Fiadeiro, S. Dustdar, & A. Sheth (Eds.), *Proceedings of BPM 2006,* Vienna, Austria, (Vol. 4102, pp. 348-357). Springer-Verlag.

Mendling, J., Moser, M., Neumann, G., Verbeek, H. M. W., van Dongen, B. F., & van der Aalst, W. M. P. (2006a). *A quantitative analysis of faulty EPCs in the SAP reference model*. BPM Center Report (BPM-06-08, BPMCenter.org).

Mendling, J., Moser, M., Neumann, G., Verbeek, H. M. W., van Dongen, B. F., & van der Aalst, W. M. P. (2006b). Faulty EPCs in the SAP rReference model. In J. L. Fiadeiro, S. Dustdar, & A. Sheth (Eds.), *Proceedings of BPM 2006,* Vienna, Austria, (Vol. 4102, pp. 451-457). Springer-Verlag.

Mendling, J., zur Muehlen, M., & Price, A. (2005). *Process aware information systems: Bridging people and software through process technology*. Wiley Publishing.

Mendling, J., Neumann, G., & Nüttgens, M. (2005a). *Workflow handbook 2005*. Lighthouse Point, FL, USA: Future Strategies Inc.

Mendling, J., Neumann, G., & Nüttgens, M. (2005b). Towards workflow pattern support of event-driven process chains (EPC). In M. Nüttgens & J. Mendling (Eds.), *Proceedings of the 2nd GI Workshop XML4BPM—XML for Business Process Management at the 11th GI Conference BTW 2005*, Karlsruhe, Germany, (Vol. 145, pp. 23-38).

Mendling, J., Neumann, G., & Nüttgens, M. (2005c). Yet another event-driven process chain. In *Proceedings of BPM 2005*, (Vol. 3649).

Mendling, J., Nüttgens, M., & Neumann, G. (2004). A comparison of XML interchange formats for business process modelling. In F. Feltz, A. Oberweis, & B. Otjacques (Eds.), *Proceedings of EMISA 2004—Information Systems in E-Business and E-Government* (Vol. 56).

Mendling, J., & Recker, J. (2007). Extending the discussion of model quality: Why clarity and completeness may not be enough. In *Proceedings of the CAiSE Workshops at the 19th Conference on Advanced Information Systems Engineering (CAiSE 2007).*

Moody, D. L. (2005). Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data & Knowledge Engineering, 55*(3), 243-276.

Moody, D. L., Sindre, G., Brasethvik, T., & Sølvberg, A. (2002, October 7-11). Evaluating the quality of process models: Empirical testing of a quality framework. In S. Spaccapietra, S. T. March, & Y. Kambayashi (Eds.), *Proceedings of the Conceptual Modeling—ER 2002, 21st International Conference on Conceptual Modeling,* Tampere, Finland, *Proceedings*, (Vol. 2503, pp. 380-396). Springer.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *IEEE, 77*(4), 541-580.

Nordsieck, F. (1932). *Die Schaubildliche Erfassung und Untersuchung der Betriebsorganisation*. Organisation—Eine Schriftenreihe. C. E. Poeschel Verlag, Stuttgart.

Nordsieck, F. (1934). *Grundlagen der Organisationslehre*. C. E. Poeschel Verlag.

Noy, N. F., Fergerson, R. W., & Musen, M. A. (2000, October 2-6). The knowledge model of protégé-2000: Combining interoperability and flexibility. In R. Dieng & O. Corby (Eds.), *Proceedings of the Knowledge Acquisition, Modeling and Management, 12th International Conference, EKAW 2000,* Juan-les-Pins, France, (Vol. 1937, pp. 17-32). Springer.

Nüttgens, M., & Rump, F. J. (2002). Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In J. Desel & M. Weske (Eds.), *Proceedings of Promise 2002,* Potsdam, Germany, (Vol. 21, pp. 64-77).

Oberweis, A. (1996). An integrated approach for the specification of processes and related complex structured objects in business applications. *Decision Support Systems, 17*, 31-53.

Oberweis, A., & Sander, P. (1996). Information system behavior specification by high-level Petri nets. *ACM Transactions on Information Systems, 14*(4), 380-420.

OMG. (2002). *Meta object facility. Version 1.4*. Object Management Group.

OMG. (2004). *Unified modeling language. Version 2.0*. Object Management Group.

OMG. (2006, February). Business process modeling notation (BPMN) specification. Final adopted specification (dtc/06-02-01). Object Management Group.

Österle, H., & Gutzwiller, T. (1992). *Konzepte angewandter Analyse- und Design-Methoden. Band 1: Ein Referenz-Metamodell für die Analyse und das System-Design*. Angewandte InformationsTechnik-Verl. GmbH.

Ouyang, C., van der Aalst, W. M. P., Dumas, M., & ter Hofstede, A. H. M. (2006). *Translating BPMN to BPEL*. BPMCenter Report BPM-06-02, BPMcenter.org.

Palmer, N. (2007). *A survey of business process initiatives*. BPT Report, Business Process Trends and Transformation+Innovation, January.

Petri, C. A. (1962a). *Fundamentals of a theory of asynchronous information flow.* Amsterdam: North-Holland.

Petri, C. A. (1962b). *Kommunikation mit Automaten*. Ph.D. thesis, Fakultät für Mathematik und Physik, Technische Hochschule Darmstadt, Darmstadt, Germany.

Philippi, S., & Hill, H. J. (in press). Communication support for systems engineering—process modelling and animation with April. *The Journal of Systems and Software*.

Pnueli, A. (1977). The temporal logic of programs. In *Proceedings of the 18th IEEE Annual Symposium on the Foundations of Computer Science* (pp. 46-57). Providence: IEEE Computer Society Press.

Porter, M. E. (1985). *Competitive advantage: Creating and sustaining superior performance*. New York: The Free Press.

Puhlmann, F., & Weske, M. (2006). Investigations on soundness regarding lazy activities. In S. Dustdar, J. L. Fiadeiro, & A. Sheth (Eds.), *Proceedings of the Business Process Management, 4th International Conference, BPM 2006*, (Vol. 4102, pp. 145-160). Springer-Verlag.

Reichert, M., & Dadam, P. (1998). ADEPTflex: Supporting dynamic changes of workflow without loosing control. *Journal of Intelligent Information Systems, 10*(2), 93-129.

Reisig, W., & Rozenberg, G. (Eds.). (1998). *Lectures on Petri nets I: Basic models*. Berlin: Springer-Verlag.

Rittgen, P. (2007). Negotiating models. In J. Krogstie, A. L. Opdahl, & G. Sindre (Eds.), *Proceedings of the 19th Conference on Advanced Information Systems Engineering CAiSE 2007*, Trondheim, Norway, (Vol. 4495, pp. 561-573). Springer-Verlag.

Rosemann, M. (2003). *Process management: A guide for the design of business processes*. Springer-Verlag.

Rosemann, M. (2006). Potential pitfalls of process modeling: part a. *Business Process Management Journal, 12*(2), 249-254.

Sarshar, K., & Loos, P. (2005, September 5-8). Comparing the control-flow of epc and Petri net from the end-user perspective. In W. M. P. van der Aalst, B. Benatallah, F. Casati, & F. Curbera (Eds.), *Proceedings of the Business Process Management, 3rd International Conference, BPM 2005,* Nancy, France, (Vol. 3649, pp. 434-439).

Scheer, A.-W. (1998). *ARIS—business process frameworks* (2nd ed.). Berlin: Springer.

Scheer, A.-W. (2000). *ARIS—business process modeling* (3rd ed.). Berlin: Springer.

Scholz-Reiter, B., & Stickel, E. (Eds.). (1996).*Business process modelling.* Springer-Verlag.

Schütte, R. & Rotthowe. T. (1998) The guidelines of modeling—an approach to enhance the quality in information models. In T. W. Ling, S. Ram, & M-L. Lee (Eds.), *Proceedings of the 17th International Conference on Conceptual Modeling*, Singapore, (Vol. 1507, pp. 240-254). Springer.

Searle, J. (1969). *Speech acts: An essay in the philosophy of language*. Cambridge, England: Cambridge University Press.

Simon, C. (2006). *Negotiation processes. The semantic process language and applications*. University of Koblenz: Habilitationsschrift.

Smith, A. (1776). *An inquiry into the nature and causes of the wealth of nations*. London.

Smolander, K., Lyytinen, K., Tahvanainen, V.-P., & Marttiin, P. (1991, May 13-15). Metaedit—a flexible graphical environment for methodology modelling. In R. Andersen, J. A. Bubenko Jr., & A. Sølvberg (Eds.), *Proceedings of the Advanced Information Systems Engineering, CAiSE'91,* Trondheim, Norway, (Vol. 498, pp. 168-193). Springer.

Soffer, P., & Wand, Y. (2004, June 7-11). Goal-driven analysis of process model validity. In A. Persson & J. Stirna (Eds.), *Proceedings of the Advanced Information Systems Engineering, 16th International Conference, CAiSE 2004,* Riga, Latvia, (Vol. 3084, pp. 521-535). Springer.

Sommerville, I. (2001). *Software engineering* (6th ed.). Addison-Wesley.

Stachowiak, H. (1973). *Allgemeine modelltheorie*. Springer-Verlag.

Stirna, J., Persson, A., & Sandkuhl, K. (2007). Participative enterprise modeling: Experiences and recommendations. In J. Krogstie, A. L. Opdahl, & G. Sindre (Eds.), *Proceedings of the 19th Conference on Advanced Information Systems Engineering CAiSE 2007*, Trondheim, Norway, (Vol. 4495, pp. 546-560). Springer-Verlag.

Strahringer, S. (1996). *Metamodellierung als Instrument des Methodenvergleichs. Eine Evaluierung am Beispiel objektorientierter Analysemethoden*. Shaker Verlag: Aachen.

Taylor, F. W. (1911). *The principles of scientific management*. New York and London: Harper and Brothers.

Valmari, A. (1998, September). The state explosion problem. In W. Reisig & G. Rozenberg (Eds.), *Proceedings of the Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the Volumes are Based on the Advanced Course on Petri Nets,* Dagstuhl, (Vol. 1491, pp. 429-528). Springer.

van der Aalst, W. M. P. (1997). Verification of workflow nets. In P. Azéma & G. Balbo (Eds.), *Application and theory of Petri nets 1997, Lecture notes in computer science* (vol. 1248, pp. 407-426). Springer Verlag.

van der Aalst, W. M. P. (1998). The application of Petri nets to workflow management. *The Journal of Circuits, Systems and Computers, 8*(1), 21-66.

van der Aalst, W. M. P., Aldred, L., Dumas, M., & ter Hofstede, A. H. M. (2004). Design and implementation of the YAWL system. In A. Persson & J. Stirna (Eds.), *Advanced information systems engineering, Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE'04)* (Vol. 3084, pp. 142-159). Berlin: Springer-Verlag.

van der Aalst, W. M. P., & and van Hee, K. (2002). *Workflow management: Models, methods, and systems*. The MIT Press.

van der Aalst, W. M. P., & ter Hofstede, A. H. M. (2005). YAWL: Yet another workflow language. *Information Systems, 30*(4), 245-275.

van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and Parallel Databases, 14*(1), 5-51.

van der Aalst, W. M. P., & Lassen, K. B. (in press). Translating unstructured workflow processes to readable bpel: Theory and implementation. *Information and Software Technology.*

van Dongen, B. F., & Jansen-Vullers, M. H. (2005, September 5-8). Verification of SAP reference models. In W. M. P. van der Aalst, B. Benatallah, F. Casati, & F. Curbera (Eds.), *Proceedings of the Business Process Management, 3rd International Conference, BPM 2005,* Nancy, France, , (Vol. 3649, pp. 464-469).

van Dongen, B. F., van der Aalst, W. M. P., & Verbeek, H. M. W. (2005, June 13-17). Verification of EPCs: Using reduction rules and Petri nets. In O. Pastor & J. Falcão e Cunha (Eds.), *Proceedings of the Advanced Information Systems Engineering, 17th International Conference, CAiSE 2005,* Porto, Portugal, (Vol. 3520, pp. 372-386). Springer.

van Hee, K., Sidorova, N., Somers, L., & Voorhoeve, M. (2006). Consistency in model integration. *Data & Knowledge Engineering, 56*.

Verbeek, H. M. W., Basten, T., & van der Aalst, W. M. P. (2001). Diagnosing workflow processes using Woflan. *The Computer Journal, 44*(4), 246-279.

Wand, Y., & Weber, R. (1990). *Studies in Bunge's treatise on basic philosophy*. Rodopi: The Poznan Studies in the Philosophy of the Sciences and the Humanities.

Wand, Y., & Weber, R. (1995). On the deep structure of information systems. *Information Systems Journal, 5*, 203-223.

Wand, Y., & Weber, R. (2002). Research commentary: Information systems and conceptual modeling—a research agenda. *Information Systems Research, 13*(4), 363-376.

Weber, I., Haller, J., & Mülle, J. A. (2006, February). Derivation of executable business processes from choreographies in virtual organizations. In *Proceedings of XML4BPM*.

Winograd, T. (1987-88). A language/action perspective on the design of cooperative work. *Human-Computer Interaction, 3*(1), 3-30.

Winter, A., Kullbach, B., & Riediger, V. (2001). An overview of the GXL graph exchange language. In S. Diehl (Ed.), *Proceedings of the Software Visualization—International Seminar,* Dagstuhl Castle, (Vol. 2269, pp. 324-336).

Wohed, P., van der Aalst, W. M. P., Dumas, M., ter Hofstede, A. H. M., & Russell, N. (2005). Pattern-based analysis of the control-flow perspective of UML activity diagrams. In L. Delcambre, C. Kop, H. C. Mayr, J. Mylopoulos, & O. Pastor (Eds.), *Proceedings of the 24nd International Conference on Conceptual Modeling ER 2005*, (Vol. 3716, pp. 63-78). Berlin: Springer-Verlag.

Wohed, P., van der Aalst, W. M. P., Dumas, M., ter Hofstede, A. H. M., & Russell, N. (2006). On the suitability of BPMN for business process modelling. In S. Dustdar, J. L. Fiadeiro, & A. Sheth (Eds.), *Proceedings of the Business Process Management, 4th International Conference, BPM 2006*, (Vol. 4102, pp. 161-176). Springer-Verlag.

Workflow Management Coalition. (2005, October 3). *Workflow process definition interface—XML process definition language* (Document Number WFMC-TC-1025). Version 2.00, Workflow Management Coalition.

Wynn, M. T., Edmond, D., van der Aalst, W. M. P., & ter Hofstede, A. H. M. (2005). Achieving a general, formal and decidable approach to the OR-join in workflow using reset nets. In G. Ciardo & P. Darondeau (Eds.), *Proceedings of the Applications and Theory of Petri Nets 2005*, (Vol. 3536*S*, pp. 423-443). Berlin: Springer-Verlag.

Wynn, M. T., Verbeek, H. M. W., van der Aalst, W. M. P., ter Hofstede, A. H. M., & Edmond, D. (2006). *Reduction rules for YAWL workflow nets with cancellation regions and or-joins*. BPMCenter Report BPM-06-24, BPMcenter.org.

Zhao, W., Hauser, R., Bhattacharya, K., Bryant, B. R., & Cao, F. (2006). Compiling business processes: untangling unstructured loops in irreducible flow graphs. *Int. Journal of Web and Grid Services, 2*(1), 68-91.

Zisman, M. D. (1977). *Representation, specification and automation of office procedures*. PhD thesis, University of Pennsylvania, Warton School of Business.

Zisman, M. D. (1978). Use of production systems for modeling asynchronous concurrent processes. *Pattern-Directed Inference Systems*, 53-68.

Zukunft, O., & Rump, F. J. (1996). *Business process modelling*. Springer-Verlag.

zur Muehlen, M. (2004). *Workflow-based process controlling. Foundation, design, and implementation of workflow-driven process information systems*. Logos, Berlin.

zur Muehlen, M., & Rosemann, M. (2004). Multi-paradigm process management. In *Proceedings of the Fifth Workshop on Business Process Modeling, Development, and Support—CAiSE Workshops*.

KEY TERMS

BPMN: BPMN is an OMG standard for modeling business processes. It is meant to be used in conjunction of BPEL as an execution language. BPMN offers different kind of task and event nodes as well as gateways for the definition of split and join conditions.

BPEL: BPEL is an XML-based language that models a business process as a composition from a set of elementary Web services. A so-called BPEL engine is a dedicated software component that is able to execute BPEL process definitions. Six of BPEL's most important concepts are briefly presented in the following, that is, partner links, variables, correlation, basic activities, structured activities, and handlers.

Business Process: "A process is a completely closed, timely, and logical sequence of activities which are required to work on a process-oriented business object. Such a process-oriented object can be, for example, an invoice, a purchase order, or a specimen. A business process is a special process that is directed by the business objectives of a company and by the business environment. Essential features of a business process are interfaces to the business partners of the company (e.g., customers, suppliers)." (Becker and Kugeler, 2003)

Business Process Model: A business process model is the result of mapping a business process. This business process can be either a real-world business process as perceived by a modeller, or a business process conceptualized by a modeller.

Business Process Modeling: Business process modeling is the human activity of creating a business process model. Business process modelling involves an abstraction from the real-world business process, because it serves a certain modeling purpose. Therefore, only those aspects relevant to the modeling purpose are included in the process model.

Business Process Modeling Language: Business process modeling languages guide the procedure of business process modeling by offering a predefined set of elements and relationships for the modeling of business processes. A business process modeling language can be specified using a metamodel. In conjunction with a respective method, it establishes a business process modeling technique.

EPCs: EPCs are a language for conceptual modeling of business processes. EPCs have three node types: functions, events, and connectors. Connectors can be used to define split and join conditions of type AND, XOR, and OR.

Petri Nets: Petri nets are a formalism that can be used for modeling business processes. A Petri net is a special kind of graph that has two types of nodes: transitions for representing business activities and routing, and places for defining pre-conditions and post-conditions of transitions.

UML Activity Diagrams: UML activity diagrams belong to the UML family of diagrams. They are used for the description of system behavior. A UML activity diagram basically consists of actions and control nodes. There are different types of control nodes for the definition of complex routing including decision, merge, fork, and join. Furthermore, UML activity diagrams offer special actions to represent the sending of a signal and receiving an event.

YAWL: YAWL is a workflow modeling language that was defined to provide straight-forward support for the 20 workflow patterns identified by Aalst et al. (2003). In addition to Petri nets and EPCs, YAWL offers constructs for the specification of multiple instance tasks and for cancellation.

ENDNOTES

1 The authors refer to the GIGA group who originally introduced the scheme.

2 Note that zur Muehlen (2004) considers simulation as a separate activity related to evaluation, but this neglects the fact that simulation is always done to evaluate different design alternatives.

3 Positivism is the philosophical theory that establishes sensual experience as the single object of human knowledge.

4 In contrast to positivism, constructivism regards all knowledge as constructed. Therefore, there is nothing like objective knowledge or reality.

5 This negation of a theoretical foundation of a modeling language has some similarities with approaches that emphasize that models are not mappings from the real world, but products of negotiations between different stakeholders, as in Hirschheim and Klein (1989) and Simon (2006).

6 Several authors use heterogeneous terminology to refer to modeling techniques. Our concept of a modeling language is similar to grammar in Wand and Weber (2002, 1990, 1995), who also use the term method with the same meaning. In Karagiannis and Kühn (2002), a modeling method is called "procedure" while the term "method" is used to define a composition of modeling technique plus related algorithms.

7 Instead of model, Wand and Weber (2002, 1990, 1995) use the term "script."

8 Implicit termination is an exception. On purpose, this pattern is not supported in order to force the modeler to carefully consider the completion condition of the process.

9 See http:www.wfmc.orgstandardsdocsStds_diagram.pdf for an overview of WfMC workflow standards and associated documents.

10 The old acronym is *BPEL4WS* (business process execution language for Web services), the new one *WSBPEL* (Web service business process execution language). The acronym *BPEL* can be used for both.

11 The committee specification is available at http:docs.oasis-open.orgwsbpel2.0CS01wsbpel-v2.0-CS01.html.

12 Basic activities in BPEL 2 are: receive, invoke, reply, assign, validate, empty, throw, rethrow, exit, wait, compensate, compensateScope, and extensionActivity.

13 ftp:www6.software.ibm.comsoftwaredeveloperlibraryws-bpelj.pdf

14 ftp:www6.software.ibm.comsoftwaredeveloperlibraryws-bpel4people.pdf

[15] https:www.sdn.sap.comirjservletprtportal-prtrootdocslibraryuuid5cbf3ac6-0601-0010-25ae-ccb3dba1ef47

[16] WSBPEL technical committee participants at OASIS are Adobe, BEA, EDS, Hewlett-Packard, IBM, IONA, JBoss, Microsoft, NEC, Oracle, Sterling Commerce, Sun, Tibco, and webMethods