Enterprise Information Systems
Vol. 00, No. 00, Month 200x, 1–25

The business process literature has proposed a multitude of business process modeling approaches or paradigms, each in response to a different business process type with a unique set of requirements. Two polar paradigms, i.e. the imperative and the declarative paradigm, appear to define the extreme positions on the paradigm spectrum. While imperative approaches focus on explicitly defining how an organizational goal should be reached, the declarative approaches focus on the directives, policies and regulations restricting the potential ways to achieve the organizational goal. In between, a variety of hybrid-paradigms can be distinguished, e.g. the advanced and adaptive case management.

This paper focuses on the less exposed declarative approach on process modeling. An outline of the declarative process modeling and the modeling approaches is presented, followed by an overview of the observed declarative process modeling principles and an evaluation of the declarative process modeling approaches.

Keywords: business process modeling, business process, declarative process modeling, process modeling languages, workflow management

ISSN: 1751-7575 print/ISSN 1751-7583 online
 DOI:
10.1080/1751757YYxxxxxxxx
http://www.informaworld.com
Enterprise Information Systems
Vol. 00, No. 00, Month 200x, 2–25

RESEARCH ARTICLE

Declarative Business Process Modeling: Principles and Modeling Languages

(*Received 00 Month 200x; final version received 00 Month 200x*)

1. Introduction

The main reason for the existence of a typical enterprise is its ability to create value for its stakeholders. More than ever, organizations are confronted with a (dynamic) environment that is highly characterized by fierce competition and high customer demands. Consequently, these organizations try to streamline their business operations by improving both the process efficiency and effectiveness. They are looking for an optimal alignment with the requirements imposed by the dynamic environment. Accordingly, business process modeling and compliance research has recently gained a lot of attention (e.g. (Mendling et al. 2010, Recker et al. 2009, Muehlen and Recker 2008), by focusing on providing an abstract description of the (cross-functional) operations involving humans, organizations (e.g. customers and suppliers), applications, documents and other operational elements. One of the approaches is declarative business process modeling.

The goal of this paper is to provide a review of the declarative process modeling research by outlining the general principles and their impact, and by discussing contemporary declarative process modeling approaches. The paper is organized as follows. In section 2 we provide a brief overview of the business process modeling research and paradigms. Section 3 presents some influential declarative process modeling approaches and briefly discusses the underlying differences between the approaches, followed by a discussion of the observed principles in section 4. Section 5 evaluates the declarative process modeling approaches and section 6 concludes the paper.

2. A Brief Overview of the Related Business Process Modeling Research & Paradigms

A business process is an abstraction of related long-running business interactions between agents, that is constituted by a specific ordering of related work activities across time and place, with a set of business goals and a set of physical and informational inputs and outputs. A multitude of business process modeling contributions have been proposed in the literature. In this section we start with defining the two main (and opposite) modeling paradigms, i.e. declarative and imperative process modeling. Next, we provide an informal introduction to declarative process modeling. Finally, the traditional business process comparison characteristics are introduced.

Note that in the literature the terms workflow and business process are often treated as synonyms. However, during the 1990s, the term business process was used in management theory, whereas the term workflow was used to denote the automated support of business processes (Rolstad˚as 1995). For the purpose of this paper, they are considered synonyms.

2.1. *Defining the Declarative and Imperative Process Modeling Paradigms*

The business process modeling research has proposed a wide variety of approaches with, at the extremes of the spectrum, the declarative and the imperative modeling paradigms.

Declarative process modeling approaches focus on what should be done in order to achieve business goals, without prescribing how an end state should be reached. Therefore, these models specify a set of constraints, business rules, event conditions or other (logical) expressions that define properties of and dependencies between activities in a business process. Consequently, all alternative paths are implicitly specified and defined as the paths that do not violate the business rules. A wide variety of declarative modeling approaches has been specified in business process management, from basic ECA-rules (Kappel et al. 1998) to the declarative process modeling languages ConDec (Pesic and van der Aalst 2006a) and BPCN (Lu et al. 2009). Note that ECA rules can be perceived as imperative on their own, as they can strictly delineate how a process should proceed. However, the complete set of ECA rules representing the process model, does not need to specify all possible transitions. Consequently, these rules put restrictions on what the final execution path should look like, but the process model may remain underspecified.

Imperative process modeling approaches focus on providing a precise definition of the control-flow of the business process in a graph-based process modeling language. These approaches are commonly used in process modeling nowadays. The basic constructs of graph-based process modeling languages are activities and control-flow dependencies between them, represented as nodes and directed arcs respectively. Several graph-based process modeling languages offer a set of additional constructs, e.g. events, data objects or compensation associations. A multitude of graph-based process modeling languages have been presented; among others Petri Net (based) modeling (Zisman 1977, Ellis and Nutt 1993), BPMN (OMG 2006), UML Activity Diagram (OMG 2004) and EPC (Keller et al. 1992).

Table 1. Concerns that affect processes.

Concern	Definition
regulations	Externally imposed directives such as among others legal requirements, standards, and contracts.
policies	Internally defined directives involving among others business strategies, tactics, and operational procedures.
benefits and costs	The incurred benefits, and costs of an activity.
time constraints	Concerns about concurrency, synchronization, due dates, and durations.
resource constraints	Capacity and availability constraints of the resources that carry out activities.
information prerequisites	The information required to make decisions.
non-functional requirements	Technical requirements such as throughput, and response time.
common-sense constraints	Common-sense constraints, such as the law of physics.

Recently, some hybrid process modeling approaches, combining both declarative and imperative constructs, have been been presented. Amongst the hybrid approaches we find the 'flexibility as a service' approaches (Van der Aalst et al. 2009) and the process materialization approach (Kumar and Yao 2009). A detailed overview of hybrid process modeling approaches can be found in (Caron and Vanthienen 2011).

In order to provide adequate support for a company's operations, information systems must be provided with sufficient information to adequately deal with the peculiarities of every-day situations. A selection of the potential real-life business concerns can be found in table 1. In the remainder of this text we will contrast the declarative aspects with their imperative counterparts.

Note that process models come under different forms, i.e. process models for information system design, executable process models and (discovered) models for process analysis. In this contribution we will primarily focus on business process models that are usually drafted during the design of an information system or process improvement.

2.2. *Introducing Declarative Process Modeling*

Each business process can be declaratively modeled by describing its state space and the set of business rules that constrain the movements in this state space (Goedertier et al. 2007, Pesic and van der Aalst 2006b).

A **state space** is a description of a discrete set of relevant states of a business process in terms of the entity types that occur in that state space. Furthermore, a state space can be characterized by one start state and possibly multiple end states. Where a **state** is considered to be a specific configuration of the facts about entities (e.g. activities) in a state space corresponding to a specific situation of a business process. The **business rules** are logical assertions about an abstract state, i.e. a logical description that specifies a set of states, that is either a necessity, an obligation, a prohibition or a possibility.

The possible movements within a business process' state space can be described by **activity state transitions**. An activity state transition represents a change in the life cycle of an activity in a business process. In (Goedertier 2008) we identified the following twelve activity transitions: create, schedule, assign, revoke, start, addFact, removeFact, updateFact, complete, abort, skip and redo, see figure 1. Such transitions involve, among others the creation, start and completion of an activity. State transitions are only considered possible if no (mandatory) business rule is violated at that point in time. A

process rule is an operationalization of a business rule that indicates when a business rule must be evaluated and enforced. It is specified for a particular type of activity state transition. For example, the business rule "the person[1] that applies for a loan must be different from the person[2] that approves the loan" can be operationalized with the process rule "On IsAssigned (person[2] approvesLoan) if person[1] is equal to person[2] then notify" (De Roover and Vanthienen 2011).

To illustrate declarative process modeling, a simplified credit approval process will be described in the remainder of the section. As stipulated in a document of the Austrian National Bank, the credit approval process requires a collaboration between the sales and the risk department of a financial institution Oesterreichische Nationalbank (OeNB) (2006). This process can be declaratively specified as:

- **state space**: the state space of the credit approval process is based on the **activity types** *handleCreditApplication, applyForCredit, checkDebt, checkIncome, reviewCredit, rejectCredit, makeProposal, rejectProposal, acceptProposal, reviewCollateral, changeApllication, collectInformation, completeContract, closeApplication*, the **entity types** *agent, customer, creditApplication*, and the state determining **business fact types** *creditApplication has beneficiary, creditApplication has collateralType, creditApplication has duration, creditApplication has amount, Customer has income, ….*
- **business rule 1**: "When the customer applies for credit the bank *should* either make a credit proposal or reject the credit application within 10 days."
- **business rule 2**: "A credit review and a collateral review are required before either a proposal is made or the credit is rejected."

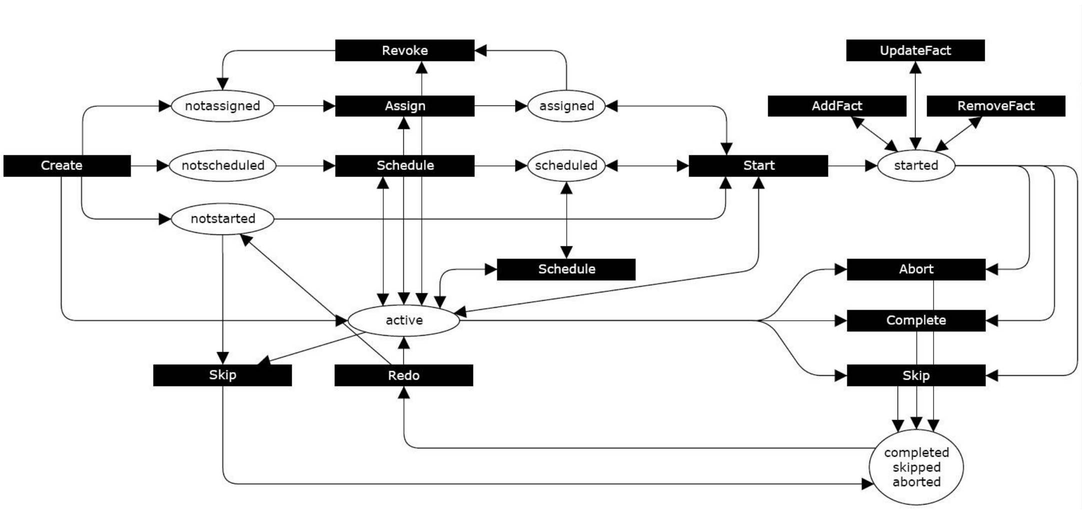


Figure 1. Activity Life Cycle in EM-BRA²CE

- **business rule 3**: "Income and debt information are a prerequisite to reviewing a credit."
- **business rule 4**: "The worker who reviews the credit cannot be the applicant of the credit application."
- **business rule 5**: "When the bank makes a credit proposal the bank is committed to complete the contract when the customer accepts the proposal within 5 days."

- **business rule 6**: “The activities check income and check debt may only be performed after a customer has applied for credit.”

The example lists a representative set of seven business rules. However, the complete credit approval process contains a far larger set of business rules. There exist powerful visualizations for the different perspectives, e.g. the control-flow can be clearly depicted with, for example, ConDec (Pesic and van der Aalst 2006b). As will become clear in the overview in section 3, no single knowledge representation language is expressive enough to cover all possible business constructs (including organizational and data elements related to the process) in process modeling. Consequently, declarative process modeling will have to rely on many different knowledge representation paradigms.

2.3. *Process Modeling Comparison Criteria*

The different process modeling approaches have a different impact on the overall performance of the business processes. From the business process modeling and reengineering/redesigning literature we derive four performance characteristics that can be used as comparison criteria (Davenport 1993, Hammer and Champy 1993, Mansar and Reijers 2005), i.e. process flexibility, compliance, efficiency and effectiveness. The purpose of these comparison criteria is to provide a general insight. An in-depth analysis of the mechanisms used in the different process modeling languages is beyond the scope of this contribution.

- **Process flexibility** is the extent to which an organization can deal with business process change occurring both at design-time and at run-time. Socio-economic factors like globalization, outsourcing, mergers and acquisitions have made business environments more complex and prone to change. In such a setting, organizations must be able to flexibly adapt their business policy and business processes to accommodate new market situations (Neumann 2009, Wang and Wang 2006, Austin and Devin 2009). In (Regev et al. 2006), a taxonomy of flexibility in business processes, based on the criteria of change, is suggested. Schonenberg et al. identify four distinct types of process flexibility in (Schonenberg et al. 2008b), i.e. flexibility by design, by deviation, by underspecification and by change. Likewise, van der Aalst and Jablonski (van der Aalst and Jablonski 2000) provide a taxonomy of change, suggest solutions, and discuss open problems.
- **Process compliance** is the extent to which a process is in correspondence with business rules, all the internally defined business constraints, and business regulations, all the externally imposed business constraints. Recently organizations are confronted with an increasing number of regulators imposing regulations that potentially affect every process within their organization. The Sarbanes- Oxley Act, for instance, not only has a substantial impact on business processes such as accounting but also on IT processes such as access management and software release management (O’Conor 2005). In general, compliance to internal policies and external regulations can be an important driver for automating business process support (Sadiq and Indulska 2011).
- **Process effectiveness** is the extent to which a business process realizes its business goals. The business goals of different business processes can be diverse. In order

processing, for example, processing the order might get the job done, whereas in an environment for collaborative product development nothing less than innovation is the main driver.

- **Process efficiency** is the extent to which the organization of the business process is capable of minimizing the amount of utilized resources such as personnel, materials, time, machine capacity. A process is cost-efficient when there is no manner of organizing work that results in better cost characteristics (e.g. in terms of the total cost of ownership). A process is time-efficient when there does not exist a way to organize the process that would result in better time characteristics (e.g. in terms of average or variability in lead time).

Of the aforementioned performance criteria, efficiency and effectiveness are usually the most crucial (Rhee et al. 2010, Xiao and Zheng 2010), as they have a direct impact on the business and financial results of an organization. However, they are to some extent compromised by process flexibility and compliance. For example as new regulations often introduce additional controls and tasks, the previously most cost- and time-efficient process path may become non-compliant. Ensuring both compliance and flexibility becomes increasingly important in a modern business environment.

The process modeling language research commonly considers two additional characteristics as well, i.e. expressibility and comprehensibility.

- The **expressibility** of a process modeling language is determined by its ability to express specific process elements, e.g. control-flow, data, execution and temporal information (Lu and Sadiq 2007)
- The level of **comprehensibility** reflects the ability of a process modeling language to define understandable process models that can be easily communicated among various stakeholders (Fahland et al. 2009a).

A full evaluation of the declarative process modeling paradigm based on the previously presented comparison criteria, can be found in section 5.

3. Declarative Process Modeling Approaches

This section will discuss a number of distinct business process modeling approaches, followed by a discussion of the differences between the approaches.

3.1. *An Overview of Declarative Process Modeling Approaches*

A common idea of declarative business process modeling is that a process is seen as a *trajectory* in a *state space* and that constraints (or business rules) are used to define the valid movements in that state space. Table 3 provides a summary of a number of business process modeling approaches, as they were presented in the literature, each with distinct declarative specifications. The differences between declarative process modeling approaches can in part be brought back to a different perception of the *state space*, the *transition types*, and the *(transition) constraints*. The selection contains both longrunning lines of research and ideas form sets of visionary papers, in order to cover a wide spectrum of declarative process modeling approaches.

Table 2. A chronological overview of process modeling languages with distinct declarative specifications

Reference	State Space	Summary	Substance of Research
ADEPT(flex) (Reichert and Dadam 1998)	data object state, activity state	The freedom of choice to change the process model of the process instance at runtime, while preserving control flow and data flow consistency regarding the addition, deletion and movement of tasks.	Long-running research
case handling (van der Aalst et al. 2005)	data object state, activity state	The freedom of choice to complete, skip, and redo activities within a number of preconditions and postconditions (hard constraints) involving the completion of preceding activities and the data object state.	Long-running research
constraint specification (Sadiq et al. 2005)	activity trace	A constraint specification framework with order, fork, serial, exclusion, and inclusion constraints (hard constraints) over a state space composed of activity traces that can occur within a subprocess (a pocket of flexibility).	Specific research
(Ferreira and Ferreira 2006)	proposition state	An integrated life cycle of the planning, user-feedback, and automated learning of the process logic of activities, represented in terms of their preconditions, and effects on case data with firstorder logic.	Specific rese arch with significant related work
ConDec (Pesic and van der Aalst 2006)	event history	A template language for Linear Temporal Logic (LTL) that describes the temporal relationships that must be observed (hard constraints) by the activities in a process.	Long-running research
PENELOPE (Goedertier and Van-thienen 2006)	event histo ry, time	Allows the modeling of the obligations and permissions that arise from (not) performing activities within specific due dates.	Specific research
Artifactcentric modeling (Bhattacharya et al. 2007)	artifact state	The modeling of the preconditions and effects on artifacts by executing services and the specification of business rules that specify when particular services are to be invoked.	Specific research
Business Process Constraint Network (Lu et al. 2009)	event history	A declarative modeling language with a graphical notation for selection and scheduling constraints.	Specific research

- Reichert and Dadam (Reichert and Dadam 1998, Dadam and Reichert 2009) describe the rationale of the **ADEPT(flex)** workflow management system (WfMS) in which end-users can change the process model of the process instance at runtime. ADEPT(flex) provides extensive user support to prevent non-permissible structural changes. In particular ADEPT(flex) preserves control flow and data flow consistency regarding the addition, deletion and movement of tasks.
- van der Aalst et al. (van der Aalst et al. 2005) describe the formal semantics of the **case handling paradigm**. Case handling is one of the few declarative modeling approaches that originates from commercial workflow management systems (WfMSs), in particular, the FLOWer WfMS of Pallas Athena. It provides the user with the freedom of choice to complete, skip, and redo activities within a number of constraints based on availability of case data and the executed activities. The state space of the case handling paradigm comprises the state of case data objects and activities. Furthermore, the system consists of data transition types (such as define, and confirm) and activity transitions types (such as complete, skip, and redo). Although there is still a preferred or normal control-flow defined between the activities, much of the

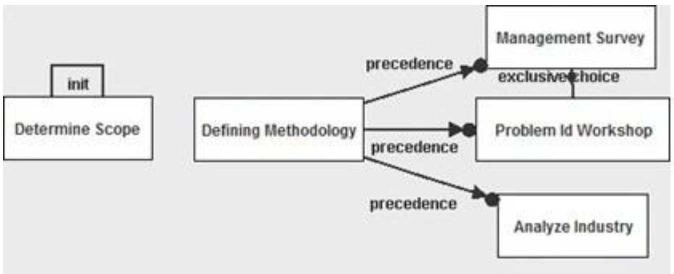
semantics of a case handling model resides in the mandatory constraint. For each activity definition, a modeler must indicate whether particular data objects are mandatory in order to be able to complete the activity.

- In the **constraint specification framework** of Sadiq et al. (Sadiq et al. 2005), order, fork, sequence, exclusion, and inclusion constraints can be specified in a state space composed of activity traces. The authors also show how it can be advantageous to combine both declarative and imperative aspects in process models. They present a foundation set of constraints for partial process modeling. A process model can contain, in addition to pre-defined activities and control flow, several so-called *pockets of flexibility*. Such pockets consist of activities, sub-processes and so-called order and inclusion constraints. Each time during enactment when a pocket of flexibility is encountered, the elicitation of the work within the pocket is done by a human end-user through a so-called "build" activity. During this build activity, the end-user constructs an instance template, a process specification, that satisfies the constraints set of the pocket of flexibility. The authors describe verification techniques for detecting redundant and conflicting constraints in the constraint set. The language has been implemented as a part of the WfMS prototype Chameleon.

- Ferreira and Ferreira (Ferreira and Ferreira 2005, 2006) consider an approach that aims at the **run-time learning and planning** of process models, rather than the design-time modeling. The state of a process consists of propositions about concepts. Performing an activity transforms the system from one particular state to another, so the transition type in the language is the act of performing an activity. The language allows to specify the preconditions of activities in terms of first-order logic. Preconditions are transitions constraints within the language: an activity can only be performed when its preconditions are satisfied. Furthermore, the effects of performing an activity can be described in terms of first-order literals that describe the sets of propositions that are either added (add-list) or removed (remove-lists) from the previous state. Rather than requiring the design-time specification of the preconditions and effects of activities, the authors propose the use of inductive logic programming techniques to discover these specifications at run-time. A partial-order planner is used to suggest possible execution plans, based on the current knowledge about preconditions and effects, that can either be accepted or rejected by the end-user. The flexibility of the approach stems from the fact that it does not require the design-time specification of an overly restrictive process model, and the run-time suggestion of alternative execution plans. The use of both planning and learning artificial intelligence techniques for business process management has also been described in e.g. (R-Moreno et al. 2000, Madhusudan et al. 2004, Jarvis et al. 1999)

- Pesic and van der Aalst (Pesic and van der Aalst 2006) and Pesic et al. (Pesic et al. 2007b) propose the declarative language **ConDec** for modeling, enacting, and verifying declarative process specifications. Enacting a ConDec process model generates a trace of events, called the event history. The state space of the language can be seen as the set of all possible event histories. Each time an activity is performed, this is recorded as an event in the event history, so performing activities is the transition type considered by the ConDec language. The language allows specifying constraint types that are defined as constraint templates in Linear Temporal Logic (LTL), i.e. configurable templates that allow the specification of a process characteristic.

Some of constraint types included are: existence constraints, relation constraints, and negation constraints. Existence constraints are activity cardinality constraints that specify how many times an activity of a particular type can be executed in a given process instance. Relation constraints are activity order constraints that specify the ordering between activity types and their existence dependencies. Negation constraints are activity exclusion constraints that specify that the occurrence of activities of some activity type excludes others. Some constraint types must be satisfied prior to the execution of a particular activity, whereas other constraints must only be satisfied upon termination of the process instance. A ConDec process model is a combination of LTL expressions that can be converted into a Bu¨chi automaton, useful for the enactment and verification of the system. Figure 2 provides an example of a consulting process modeled with ConDec and some related LTL statements. Three control-flow constraint types have been used in this example: the initial activity constraint for "determine scope", the activity precendence constraint (e.g. a management survey may not start before the methodology has been defined) and the exclusive choice constraint (e.g. "either a management survey" or a "problem identification workshop" will be performed). A ConDec process model can be verified by checking whether the model contains dead activities or conflicting constraints. The ConDec language is part of the DECLARE WfMS prototype (Pesic et al. 2007a).



$$(\neg ManagementSurvey) W (DefiningMethodology)$$
$$(\neg AnalyzeIndustry) W (DefiningMethodology)$$

Figure 2. Consulting Process in ConDec Modeling Notation (ConDec (Pesic and van der Aalst 2006))

- Several authors describe languages for intelligent agents to reason about contract state (Mar´ın and Sartor 1999, Yolum 2005, Knottenbelt and Clark 2004, Governatori 2005, Paschke and Bichler 2005, Haq et al. 2009). Contracts represent binding commitments between two or more parties and are important business concerns governing business processes. Contract modeling is akin to declarative process modeling. In Goedertier and Vanthienen (Goedertier and Vanthienen 2006) a language is defined for modeling a contract in terms of a set of so-called temporal deontic assignment rules. This language is the **PENELOPE** language. In the language, temporal deontic assignments are obligations and permissions of agents to perform a particular activity within an indicated deadline. The existence of temporal deontic assignments depends on earlier performed activities and the system time. Examples of deontic assignments are: "initially the buyer has the permission to place an order" and "when the buyer places an order, the seller must either accept or reject it within one time period". Therefore, the state space of the PENELOPE language consists of the event history and the system time. The language considers two transition types: performing activities and deadline violations. Transitions in the language are constrained by the requirement that obligations and permissions should not be violated (hard constraint). The specification of a set of temporal deontic rules can quickly become incomprehensible. Therefore, it is indicated how, under a number of limitations, a set of temporal deontic rules can be visualized in a graph-oriented process modeling language. Furthermore, it is indicated how the verification of temporal deontic assignments could be performed.

- Bhattacharya et al (Bhattacharya et al. 2007) formally describe a so-called **artifactcentric process modeling** approach, that allows to model processes in terms of the preconditions and effects of services. The state space of their language consists of artifacts. Artifacts are object-oriented data structures with attributes and states. The transition type in the language is the act of invoking a service. Services can create or destroy artifacts and can read and write their attributes. In order to invoke a service on a set of artifacts, the preconditions of the service must be fulfilled. Artifact-centric modeling also provides for the specification of the effects of services, but the authors do not propose a planning mechanism that reasons about the effect of invoking service in order to obtain a particular goal state. Instead, the system is given dynamism by means of business rules. These business rules are production rules that can either determine the conditions under which an artifact can change its state, or the conditions under which a particular service must be invoked. The modeling approach is in spirit similar to previous languages, but interestingly, the authors also present a number of complexity results concerning reachability of an end state for an artifact class, deadlock detection, and the detection of redundant attributes. In (Fritz et al. 2009) the authors propose a framework that should ultimately result in tools for the automatic construction of artifact-based business processes from the following inputs: the artifact families, the available services with their preconditions and effects and the goals that need to be satisfied. This technique constructs a maximum workflow schema containing all paths that allow every execution to both complete and satisfy the proposed goals.

- Lu et al. (Lu et al. 2009) propose **business process constraint networks (BPCN)** as an approach for declarative process modeling. Sixteen different constraint types are categorized into two categories: the selection constraints and the scheduling constraints. Selection constraints place restrictions on the activities that can be used in process instances (e.g. mandatory, cardinality, and exclusion constraints). Scheduling constraints, on the other hand, limit the execution possibilities of the selected activities both in terms of control dependencies and of temporal dependencies. The former specify the activities that must be included within the process and their ordering, e.g. sequence and parallelism. Temporal dependencies describe inter-activity time restrictions, e.g. a relative deadline such as, activity[2] must be started within x time units after the completion of activity[1]. Each of the constraint types can be expressed using constraint templates, which mask the approaches' formal semantics. A binary Boolean constraint network and a qualitative temporal constraint network are used to verify, validate and reason about the set of constraints. At run-time business process variants are constructed in a two-step way, a task selection followed by a task scheduling. In (Lu et al. 2006) the authors provide an informal and abstract description of a suitable execution framework.

3.2. *Differences between the Declarative Process Modeling Approaches*

Although there already exist many languages for declarative process modeling, these languages all are fundamentally different and none of them covers all possibilities. In this section the major differences between the declarative process languages are discussed.

- **Different business concerns.** Each of these languages only allows to model a subset of the many real-life business concerns that exist in reality. For instance, the ConDec language and the PENELOPE language only allow to express business rules about sequence and timing constraints, i.e. the control flow aspects (Heinl et al. 1999).
- **Different state space.** Because existing languages model different business concerns, they have different conceptions about the state space of a business process. Approaches such as artifact-centric process modeling and semantic web services consider the facts about business concepts to be the only discriminant of process states. The case handling considers both the data object state and the current activity state. The ConDec language perceives the event history (the trace of executed transitions), whereas PENELOPE also includes the system time, in order to take into account due dates.
- **Different constraint types.** Even when languages consider a similar state space and transition types, they have different ways of expressing transition constraints. The ConDec language for instance, expresses temporal constraints that (eventually) must hold between activities in a trace, whereas the PENELOPE language discusses business constraint types that are essentially temporal deontic assignments.
- **Different knowledge representation and reasoning paradigms.** Finally, every language uses a different ontology, different ontology language, and different languages for expressing constraints. For instance, the ConDec language makes use of Linear Temporal Logic (LTL) as underlying paradigm whereas the PENELOPE language makes use of the Event Calculus.

So future research could focus on the strong points and specific application areas of each of these approaches, or on further convergence and integration.

4. Observed Declarative Process Modeling Principles

This section aims at discussing the main declarative process modeling principles, in comparison with imperative process modeling. These principles can be observed in all major declarative languages, with some minore variations. The principles are grouped in five broad categories:

- **Modeling business concerns.** The first category of principles deals with the business concerns management aspects, i.e. the concerns' specification, adaption, enforcement and modality
- **Allowable process paths.** In the allowable process paths category we grouped the principles related to the way of specifying process paths and its impact on overspecification of process models.

- **Modeling detail.** The third category covers the principles related to the level of detail that is present in the declarative process models.
- **Process coordination.** Process coordination principles are grouped in the fourth category and cover topics such as the agent viewpoint, dealing with coordination and communication activities and the third-person perspective.
- **Model representation.** This category discusses the understandability and expressibility principles in the model representation subsection.

4.1. *Modeling Business Concerns*

4.1.1. Managing Changing Business Concerns

The business rules used in declarative process modeling approaches represent atomic formal expressions of the business concerns, as discussed in section 2.1. Declarative process models contain **explicit representations of the involved business concerns**. When the supported business process models are human-understandable and machineexecutable specifications (Recker et al. 2005, Recker 2010), process analysts are able to **trace back** the policies and regulations to the business rules that enforce them. This transparency enables an effective identification of the business rules that need to change. Verification of the new business rules sets remains an important issue. Imperative process modeling implicitly uses business concerns to determine task control flows, information flows and work allocation schemes. In (Fahland et al. 2009b) it has been suggested that implementing circumstantial changes would be easier with declarative process modeling (e.g. in ConDec modeling notation Pesic and van der Aalst (2006)), whereas implementing sequential changes would be easier with imperative process modeling.

In an ideal situation, the business rules should be specified in a common language between the business-side and IT-side of organizations. Such a language allows the business-side to formally represent models of how it operates internally and how it can legally interact with business partners (Nelson et al. 2009). This way the business remains owner of its business logic. At the same time, such a common language allows the IT-side to have information systems support business processes accordingly, with as little development effort as possible (zur Muehlen and Indulska 2010). In reality, however, we have to observe that some of the declarative languages require a vast amount of knowledge about formal logic (Lu and Sadiq 2007). Comprehensibility is therefore an important concern.

4.1.2. Separating the Business Concern Modeling from the design of Enforcement Mechanisms

Declarative process modeling separates business rule modeling from business rule enforcement. In particular, it **does not make use of the control-flow** to indicate when and how business rules are to be enforced. Instead, it is left to the execution semantics of the declarative process models to define an execution model in which different kinds of business rules are automatically enforced.

Imperative process languages predominantly focus on the control-flow perspective of business processes. In such process languages it might be possible to enforce business rules using a control-flow-based modeling construct. For instance, the enforcement of a derivation or authorization rule can be modeled as a decision shape in the Business Process Modeling Notation (BPMN), the de facto imperative modeling standard (Object Management Group 2006). Since imperative process modeling techniques do not allow an independent formulation of all business rules that are enforced in the process models (Hoffmann et al. 2009), the same business rule might be duplicated in several imperative process models. When the business rule changes it is likely that all process models must be reexamined, thereby creating additional maintenance issues.

Figure 3 represents how a threshold related authorization rule can be enforced using control-flow constructs in BPMN. An informal declarative equivalent of this process model is:

- **state space**: the state space of the credit approval process is described by the **activity types** *applyFor Credit, reviewCredit, makeProposal and rejectProposal*, the **concept types** *bank, sales, risk control, customer and creditApplication*, and the **business fact types** *creditApplication has amount, ...*
- **business rule 1**: "A credit application is required before a credit review is performed."
- **business rule 2**: "A credit review is required before either a proposal is made or the credit is rejected."
- **business rule 3**: "The bank must review the credit application."
- **business rule 4**: "If the customer applies for more than EUR 2000, then the credit application must be reviewed by risk control."
- **business rule 5**: "When the customer applies for credit the bank *should* either make a credit proposal or reject the credit application." • ...

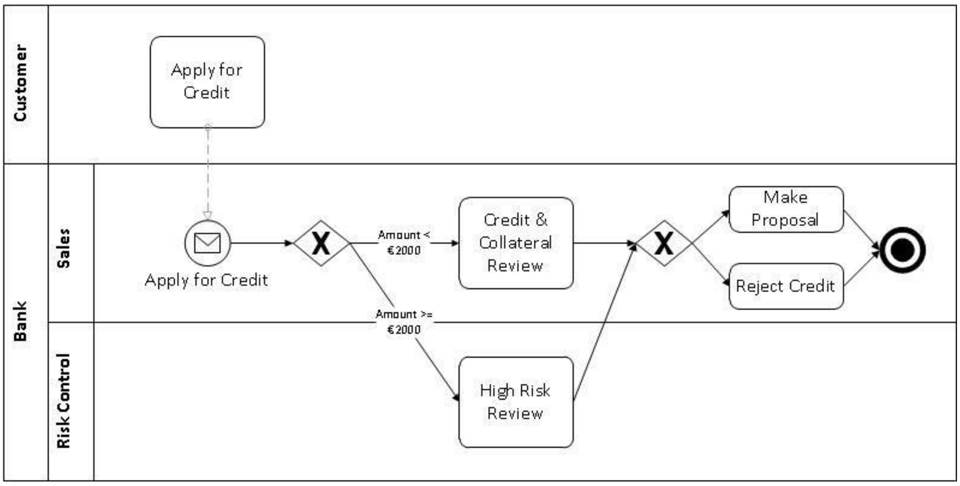


Figure 3. Imperative rule enforcement: authorization rule

4.1.3. Differentiation by Modality

Another point of difference between the imperative and declarative is the modality that is attached to the information in process models. Imperative process models inherently have the necessity modality (*must*) attached, whereas declarative process languages allow to differentiate by attaching **different modalities** like intention (*ought*), advice (*should*), possibility (*can*) and factuality (*is*) to parts of the process model. These modalities offer run-time flexibility. In particular, they allow to distinguish between what is strictly required (hard constraint) and what is merely desirable (soft constraint) behavior in a business process. This can help the agents in the business process to come up with a suitable yet valid execution plan.

The idea of different modalities is related to the research of Suchman (Suchman 1995), Schmidt and Simone (Schmidt and Simone 1996) and Ross (Ross 2003). Suchman (Suchman 1995) points out that business process models can never fully represent cooperative work in all its facets. In any organization, representations of work are required to create a common understanding of the work and thus facilitate coordination. However, workers may have conflicting views on the work. Suchman warns that a normative, prescriptive account of how the work gets done might severely differ from specific working practices. Although representations of work are a useful tool to reason about work and to steer activities, they risk becoming useless when used outside the context of the work. According to the seminal work of Suchman (Suchman 1987) representations of work need to be underspecified such that they are plans for situated action. The worker uses a plan as a guideline to go about but also determines the most suitable activity to undertake by himself from the context of the process.

4.2. *Allowable Process Paths*

4.2.1. Implicitly Specifying Process Paths

Unlike imperative process modeling, declarative process modeling does **not involve the pre-computation of task control flows**, information flows and work allocation schemes (Rychkova et al. 2008, Schonenberg et al. 2008a). Activity dependencies remain implicit in declarative process models. An explicit enumeration of all activity dependencies is not required - and often even difficult to obtain (Heinl et al. 1999). During the execution of a declarative process model, a suitable execution scenario is constructed (either by a human or machine coordinator) that realizes the business goals of the process model. The latter is called goal-driven execution and its automation is akin to planning in the domain of Artificial Intelligence (Nau et al. 2004). The results of an empirical test with the Alaska simulator (Weber et al. 2009a) indicate that end-users can effectively use this agile process planning over a considerable spectrum of constraints. For determining the validity of declarative process models, execution trajectories can still be obtained from implicit process models. Additionally, the resulting under-specified process models allow for better exception handling. This has been discussed in e.g. (Wang and Kumar 2005).

Explicitly specified flows, as can be found in imperative process models, might result in considerable process efficiency for static and standardized business processes (Lu and Sadiq 2007). Over time, several practices were proposed in order to increase the flexibility of imperative process models, such as in (Heinl et al. 1999, Yang 2004, Reijers et al. 2009, Weber et al. 2008, Doehring and Zimmermann 2011, Doehring et al. 2011). Additionally, a wide variety of analysis and verification techniques has been presented (Van Der Aalst et al. 1998).

4.2.2. Avoiding Assumption Bias

The business rules in declarative process models can be traced back to an original business concern. Consequently, declarative process models are likely only to contain a **minimum of constraints** regarding a particular business process, i.e. underspecification. This minimal constraint set ideally reflects the directives imposed on the process. Imperative process models, on the other hand, are the result of an explicit pre-computation of task dependencies (see section 4.2.1). Therefore, it is not guaranteed that imperative process models include a number of additional assumptions that might *overly specify* the underlying business process. The claim that imperative process models are often overspecified was first made by Pesic and van der Aalst (Pesic and van der Aalst 2006).

However, organizations in a knowledge-based economy will increasingly be confronted with requests that require a unique problem resolution strategy or workflow path. While basic service levels are becoming self-evident, organizations will be able to differentiate themselves from others in the way they are able to deal with specific issues (within the boundaries set by the business concerns). Case management and other declarative modeling paradigms allow for a goal-driven approach that enables the creation of customized paths and solutions in the context of unstructured processes such as incident management and claim handling.

4.3. *Modeling Detail*

4.3.1. Activity-Level Granularity

Declarative process models have a more fine-grained model granularity than imperative process models. Whereas imperative process languages are process-centric in that they model business processes, declarative process languages are activity centric, as they model the business concerns related to a set of activity types. Business process models are composed of activity types, but the same activity type can occur in multiple business process models. In addition, many business concerns range over activity types and are not specific to one business process model in particular. Therefore activity-centric models have the advantage that these governing aspects are not a-priori straitjacketed into a particular business process model. For instance, the regulation that a purchase order must never be paid prior to the reception of an invoice, can possibly be relevant in different business processes. To allow the reuse of this regulation, it must be specified across the boundaries of artificially delineated business process models.

The process-oriented view on organizations has lead to a better understanding of the value chain (Porter 1985, Davenport 1993) and has improved business process redesign. However, by letting the activity-centricity set the granularity it is observed that we could unlock some interesting possibilities. When required, a process-centric model can be obtained from an activity-centric model. It might be easier to transform an activitycentric model into a process-centric model then vice-versa, as different activity constraints may be dispersed over different process models.

4.3.2. *Taking the Complete Activity Life-cycle into Account*

Imperative process models do not explicitly consider the life cycle of the activities within a business process, but represent activities as actions that happen instantaneously. This is, for instance, the case for workflow nets (van der Aalst et al. 1994, Eshuis and Dehnert 2003). Whereas this simplifying representation might be a useful abstraction for process visualization or verification purposes, it does not take into account the fact that activities have a life cycle of their own that consists of creation, planning, execution, and exception handling events. Declarative process models, in contrast, may consider other events in the life cycle of activities such as the creation, scheduling, assignment, start, fact manipulation, completion, skipping, cancelation and redoing of an activity. An overview of the activity state transitions that were defined in the context of EM-BRACE can be found in figure 1 (Goedertier 2008). Consequently, (transition) constraints in support of the business concerns can be more fine-grained, e.g. when a buyer *completes* an application for credit activity, the bank has the obligation to *start* a review credit activity.

4.4. *Process Coordination*

4.4.1. *Human-Machine Distinction versus Agents*

Information systems and machinery have led to an extensive automation of both work and coordination work. But not all activities in every business process can be fully automated. Likewise, not every business process lends itself to the same degree of automated coordination. In many cases, some of the (coordination) work is performed by machines and some of it by humans. While imperative process models distinguish between humans and machines (Basu and Kumar 2002), declarative process models (ideally) make abstraction from the differences between humans and machines in performing (coordination) work. Rather than making an ontological distinction between concepts like humans and machines, both concepts are unified through the use of the agent metaphor (Woolridge and Wooldridge 2001). Agents can be entire organizations, organizational units or individual workers and machines. In many cases, individual agents - whether humans, machines or a combination of both - act on behalf of the organization to which they pertain. For example, a transport activity might require the scheduling of a driver, truck, and trailer that act as an ad-hoc group of agents.

4.4.2. *Coordination Work*

Business process management is about the coordination of work (Schmidt and Simone 1996, Weske et al. 2004). Imperative process models are often an explicit specification of the coordination work. In contrast, declarative process models make no distinction between coordination work and regular work. Coordination might be an activity for another actor and by not specifying strict sequences the design-time flexibility might improve. What may appear as work to an external agent, may very well be coordination work to another agent. For instance, a sales representative may instruct the expedition department to ship an order by a particular due date, but this activity may conceal the coordination of many other activities within the expedition department.

4.4.3. *Communication Logic*

Imperative process models contain a lot of communication activities intended to notify an external business partner about the occurrence of a relevant business event or to transmit information. Figure 2 represents an excerpt from the BPMN specification (Object Management Group 2006, p. 107) that contains the communication activities ‘receive order’ and ‘send invoice’. Such communication activities depict communication logic in an imperative manner, because they specify how and when business events are communicated and information is transmitted.

Declarative process models, on the other hand, are only concerned with the ability of business agents to perceive business events and business concepts. When an agent (for instance a business partner) can perceive a particular event, the event becomes non-repudiable to the agent, irrespective of how the agent is notified of the event. The execution semantics of a declarative process model determines how events are communicated. In particular, events can be communicated as messages that are sent by the producer (push model), retrieved by the consumer (pull model), or via a publish-subscribe mechanism.

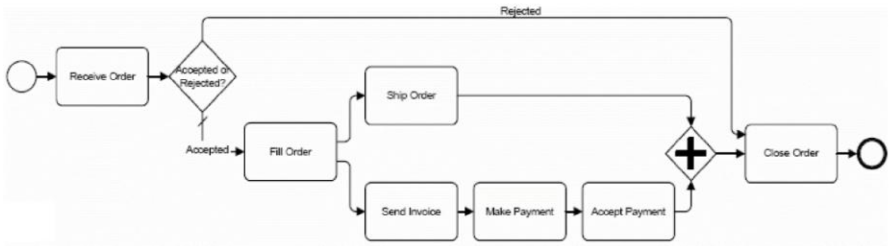


Figure 4. Communication logic in a imperative process model (Object Management Group 2006, p. 107)

4.4.4. *Third-Person Perspective*

The growing popularity of the Internet based on IP-based communication protocols and technologies such as XML, has given rise to the requirement of automated coordination of business processes across the boundaries of individual organizations. As a consequence, it is not always technically or economically viable to have processes coordinated centrally. Another consequence of distribution is that it is unlikely that process designers can come up with only one representation of work. In many cases all business partners that participate in a cooperation might have different representations of the cooperative work. These representations are to be kept in part private from other process business partners.

Declarative business process models must take into account these disparate perspectives on processes. When modeling behavior it is proposed to adopt a *third-person perspective* - what will an actor with a particular role do in response to what others do? rather than a *first-person perspective* - what will I do in response to what others do? In a third-person perspective all roles, actors and organization structures are named without the modeler adopting a particular viewpoint. A third-person modeling perspective has the advantage that it is possible to distinguish multiple interacting actors within a single organization. Another advantage is that business rules can be more easily shared in a business community when they are expressed from a third-person perspective. The modeling perspective distinction is, for instance, present in the literature about process orchestration and choreography (Bussler 2001, Peltz 2003) or in the distinction between internal and external Agent-Object-Relationship models (Wagner 2003).

4.5. *Model Representation: Understandability and Expressibility*

High levels of understandability and expressibility are attractive characteristics for business process modeling languages. The use of an imperative process modeling language with a clear syntax generally leads to comprehensible process models, when these process models are not too large (Lu and Sadiq 2007, Mendling et al. 2010). In contrast, declarative process specifications may become less readable, due to the many (interacting) constraints (Fickas 1989) or due to the use of formal modeling languages, e.g. linear temporal logic (Pesic 2008). This is especially true in the context of establishing sequential information, which can be well-organized in a imperative process model.

Research performed by Fahland et al. demonstrated that establishing domain knowledge and circumstantial information is easier with declarative modeling languages (Fahland et al. 2009a). Similar conclusions are put forward in (Lu and Sadiq 2007).

4.6. *Concluding Remarks*

Table 3 summarizes the different design principles presented in the previous sections and provides for each of these declarative principles the imperative point of view.

Table 3. Imperative versus declarative process modeling principles

	Principles	Declarative modeling	Imperative modeling	Section
Modeling business concerns	Business concerns	explicit	implicit	4.1.1
	Rule enforcement	what	what, when and how	4.1.2
	Modality	must, ought and can	must	4.1.3
Allowable process paths	Path construction	run-time	design-time	4.2.1
	Assumption bias	underspecified	overspecified	4.2.2
Modeling detail	Model granularity	activity-centric	process-centric	4.3.1
	Activity life-cycle	multi-event	single event	4.3.2
Process coordination	Coordinator/worker	agent	human-machine distinction	4.4.1
	Coordination/activity	coordination is an activity	coordination is not an activity	4.4.2
	Communication	what	what and how	4.4.3
	Perspective	third-person	first-person	4.4.4
Model representation	Understandability	lower	higher	4.5
	Expressibility	higher	lower	4.5

The previous subsection described the differences in design principles between imperative and declarative process modeling, however, **no dichotomy is implied** (Caron and Vanthienen 2011). The literature has indicated that business process modeling techniques can combine both imperative and declarative modeling aspects (Fahland et al.

2009a). Firstly, business concerns can be made traceable by carefully documenting them using a declarative modeling approach at design-time, afterwards an artificial planning technique could be used to derive a imperative process model, which would ease the execution (Maghraoui et al. 2006, Hendler et al. 1990). Secondly, design principles of both paradigms can also be combined, e.g. through the use of placeholder activities (Sadiq et al. 2005) or rule-based adaption of imperative reference process models (Kumar and Yao 2009). In this context the paradigms can be seen as complementary, as they can be used together to realize their combined advantages.

5. Evaluation of the Declarative Process Modeling Approaches

The declarative process modeling literature is characterized by its aim to **reconcile process flexibility and compliance**. Adapting the business rules to changes in the business policies or imposed regulations is straightforward, since there is no duplication of business logic in a declarative process model (nor over multiple process models). Moreover, additional business rules can be directly added, without the need to fully redesign the business process. Consequently the **design-time flexibility**, related to both process adaptability and maintainability, of a declarative business process model can be high.

Process compliance might be fully guaranteed when all relevant business policies and regulations are mapped on mandatory business rules, which results in traceability and facilitates verification by domain specialists. In order not to affect the **process effectiveness**, constraints should be valid, consistent, feasible etc. The impact on the **process efficiency** of the declarative process model can be influenced by specifying guideline constraints, which specify an optimal execution path but can be violated during execution.

Languages representing rule-based process modeling provide a **higher expressibility** than graph-based languages (e.g. the ability to specify temporal requirements) (Lu and Sadiq 2007), but might result in process models which are **less comprehensible** (Fickas 1989) due to large and possibly unstructured sets of business rules.

Declarative process models guarantee a **high run-time flexibility** for declarative process specifications that contain only the strictly required mandatory constraints. An individual execution path that satisfies the set of mandatory constraints can be dynamically built for a specific process instance. Process *compliance is assured* when all regulation and business policies are correctly mapped onto mandatory business constraints.

Flexibility comes at a price. Efficiency of executing declarative models may be a concern, as during the construction of a suitable execution path **limited support** might be provided to the end user (Weber et al. 2009b). However, in (Schmidt and Simonee 1996, Barba and Del Valle 2011) the idea of differentiating constraints by modality was proposed; soft constraints would guide the user whereas mandatory constraints would ensure compliant behavior. The guidance provided by the soft constraints might depend on explicit domain knowledge or can be learned through process mining (Schonenberg et al. 2008c). This additional guidance may result in an improved efficiency and effectiveness.

Well designed imperative process models (based on formal semantics and precise specifications), in contrast, can be executed rather efficiently and effectively (Goedertier and Vanthienen 2009). Additionally, these process models tend to be understandable. However, changing business concerns may result in maintainability issues (Fahland et al. 2009b). A large variety of **hybrid-paradigms** that combine imperative and declarative principles can be distinguished (Sinur 2009), e.g. the case driven approach of adaptive/advanced case management (Swenson and Palmer 2010), the pockets of flexibility (Sadiq et al. 2005), process materialization (Kumar and Yao 2009), etc. The use of hybrid approaches regularly results in a moderation of the impact on the process characteristics.

Dynamic, human-centric, non-standardized business processes are most likely to require the runtime flexibility offered by declarative process modeling. Examples are the handling of distress calls in call centers, insurance claim handling, incident management or the coordination of patient processes in hospitals. In contemporary knowledge-based industries the focus has shifted towards these more dynamic business processes. Many of them are being supported by case management information systems (Swenson and Palmer 2010).

6. Conclusion

Designing information systems that provide support for operational business processes with the right level of process flexibility, compliance, efficiency and effectiveness can be a challenging task. Additionally, several intrinsic tradeoffs between these process qualities exist, e.g. between compliance and flexibility.

Declarative process modeling focuses on explicitly specifying the business concerns (i.e. business policies and regulations) that govern the business process. By formulating only the minimal business concerns and leaving as much freedom as is permissable to a domain specialist at run-time, declarative process modeling techniques try to reconcile process flexibility and compliance. Imperative business process models, on the other hand, contain an explicit specification of all alternative execution paths, events and exceptions. While explicitly prescribing how a process should proceed can be beneficial for the understandability of the model, the demonstration of process compliance and the efficiency of process execution afterwards, it might severely limit process flexibility.

The choice of the modeling paradigm depends on the type of application. Business processes that are characterized by a dynamic, human-centric and non-standardized setting, will benefit from the flexibility that could potentially be provided by declarative process modeling (e.g. healthcare processes). Moreover, in declarative process modeling approaches compliance with internal and external directives might be easily demonstrated. Static, machine-centric, standardized business processes are most likely to benefit from an imperative approach as effectiveness and efficiency can be easily optimized (e.g. the processing of routine production orders or standardized financial transactions). Additionally, a case could be made for the hybrid approaches taken from the wide spectrum of modeling paradigms in between declarative and imperative approaches.

In this paper we presented an overview of different approaches to declarative process modeling, ranging from complementing imperative models with declarative logic (e.g. the ADEPT approach) to fully fledged declarative process modeling approaches (e.g. ConDec). These declarative approaches differ from each other in the business concerns that can be specified, the required state space, the available constraint types and the knowledge representation and reasoning paradigms used. However, their objectives, urge for flexibility and application types remain the same.

Additionally, the study resulted in the identification of a multitude of interesting declarative process modeling principles shared by all approaches. With the first set of principles we indicated how declarative process modeling manages business concerns. The paradigm opts for making business concerns explicit and for taking modality into account, resulting in higher flexibility, traceability and adaptability. The second set describes the absence of pre-computed control flows and its positive effect on the reduction of the assumption bias. The resulting underspecified models provide additional flexibility. Thirdly, through the use of activity-level granularity and the concept of activity life-cycle, declarative process modeling techniques are able to produce more specific transition constraints that are instantaneously transferred to all processes containing that activity. Fourthly, declarative process modeling approaches focus on agents, consider coordination work as regular tasks, avoid modeling communication logic and embrace the third-person perspective. Finally, declarative process models might tend to trade some understandability for higher expressibility.

References

Austin, R., Devin, L., 2009. Research Commentary: Weighing the Benefits and Costs of Flexibility in Making Software: Toward a Contingency Theory of the Determinants of Development Process Design. Information Systems Research 20 (3), 462–477.

Barba, I., Del Valle, C., 2011. A planning and scheduling perspective for designing business processes from declarative specifications. In: Filipe, J., Fred, A. (Eds.), Proceedings of the 3rd International Conference on Agents and Artificial Intelligence. Vol. 1. Rome, Italy, pp. 562–569.

Basu, A., Kumar, A., 2002. Research commentary: Workflow management issues in ebusiness. Information Systems Research 13 (1), 1–14.

Bhattacharya, K., Gerede, C. E., Hull, R., Liu, R., Su, J., 2007. Towards formal analysis of artifact-centric business process models. In: Business Process Management. pp. 288–304.

Bussler, C., 2001. The role of B2B protocols in inter-enterprise process execution. Technologies for E-Services, 16–29.

Caron, F., Vanthienen, J., 2011. An exploratory approach to process lifecycle transitions from a paradigm-based perspective. Enterprise, Business-Process and Information Systems Modeling, 178–185.

Dadam, P., Reichert, M., 2009. The adept project: a decade of research and development for robust and flexible process support. Computer Science-Research and Development 23 (2), 81–97.

Davenport, T., 1993. Process innovation: reengineering work through information technology. Harvard Business Press.

De Roover, W., Vanthienen, J., 2011. Unified patterns to transform business rules into an event coordination mechanism. Lecture Notes in Business Information Processing 66, 730–742.

Doehring, M., Zimmermann, B., 2011. vbpmn: Event-aware workflow variants by weaving bpmn2 and business rules. Enterprise, Business-Process and Information Systems Modeling, 332–341.

Doehring, M., Zimmermann, B., Karg, L., 2011. Flexible workflows at design-and runtime using bpmn2 adaptation patterns. In: Business Information Systems. Springer, pp. 25–36.

Ellis, C., Nutt, G., 1993. Modeling and enactment of workflow systems. Application and Theory of Petri Nets 1993, 1–16.

Eshuis, R., Dehnert, J., 2003. Reactive Petri Nets for Workflow Modeling. In: van der Aalst, W. M. P., Best, E. (Eds.), Proceedings of the 24th International Conference on Applications and Theory of Petri Nets (ICATPN 2003). Vol. 2679 of Lecture Notes in Computer Science. Springer, pp. 296–315.

Fahland, D., Lu¨bke, D., Mendling, J., Reijers, H., Weber, B., Weidlich, M., Zugal, S., 2009a. Declarative versus imperative process modeling languages: The issue of understandability. In: Enterprise, Business-Process and Information Systems Modeling: 10th International Workshop, BPMDS 2009. Springer, pp. 353–366.

Fahland, D., Mendling, J., Reijers, H., Weber, B., Weidlich, M., Zugal, S., Sep. 2009b. Declarative vs. Imperative Process Modeling Languages: The Issue of Maintainability. In: Mutschler, B., Wieringa, R., Recker, J. (Eds.), 1st International Workshop on Empirical Research in Business Process Management (ER-BPM'09). Ulm, Germany, pp. 65–76.

Ferreira, D. R., Ferreira, H. M., 2005. Learning, planning, and the life cycle of workflow management. In: Proceedings of the 9th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2005). pp. 39–46.

Ferreira, H., Ferreira, D., 2006. An integrated life cycle for workflow management based on learning and planning. International Journal of Cooperative Information Systems 15 (4), 485–505.

Fickas, S., 1989. Design issues in a rule-based system. Journal of Systems and Software 10 (2), 113–123.

Fritz, C., Hull, R., Su, J., 2009. Automatic construction of simple artifact-based business processes. In: Proceedings of the 12th International Conference on Database Theory. ACM, pp. 225–238.

Goedertier, S., September 2008. Declarative techniques for modeling and mining business processes. Phd thesis, Katholieke Universiteit Leuven, Faculty of Business and Economics, Leuven.

Goedertier, S., Haesen, R., Vanthienen, J., 2007. EM-BrA^{2}CE v0.1: A vocabulary and execution model for declarative business process modeling. FETEW Research Report KBI 0728, K.U.Leuven.

Goedertier, S., Vanthienen, J., 2006. Designing compliant business processes with obligations and permissions. In: Eder, J., Dustdar, S. (Eds.), Business Process

Management Workshops. Vol. 4103 of Lecture Notes in Computer Science. Springer, pp. 5–14.

Goedertier, S., Vanthienen, J., April 2009. An overview of declarative process modeling principles and languages. Communications of SWIN 6, 51–58.

Governatori, G., 2005. Representing business contracts in RuleML. International Journal of Cooperative Information Systems 14 (2-3), 181–216.

Hammer, M., Champy, J., 1993. Reengineering the corporation. Harper Collins, New York, NY.

Haq, I., Paschke, A., Schikuta, E., Boley, H., 2009. Rule-Based Workflow Validation of Hierarchical Service Level Agreements. In: Grid and Pervasive Computing Conference, 2009. GPC'09. Workshops at the. IEEE, pp. 96–103.

Heinl, P., Horn, S., Jablonski, S., Neeb, J., Stein, K., Teschke, M., 1999. A comprehensive approach to flexibility in workflow management systems. SIGSOFT Software Engineering Notes 24 (2), 79–88.

Hendler, J., Tate, A., Drummond, M., 1990. AI planning: Systems and techniques. AI magazine 11 (2), 61.

Hoffmann, J., Weber, I., Governatori, G., 2009. On compliance checking for clausal constraints in annotated process models. Information Systems Frontiers, Special Issue on Governance, Risk, and Compliance 43, 48.

Jarvis, P., of Edinburgh. Artificial Intelligence Applications Institute, U., of Loughborough. Department of Chemical Engineering, U., 1999. Exploiting AI technologies to realise adaptive workflow systems. AIAI, University of Edinburgh.

Kappel, G., Rausch-Schott, S., Retschitzegger, W., 1998. Coordination in workflow management systemsa rule-based approach. Coordination Technology for Collaborative Applications, 99–119.

Keller, G., Nuttgens, M., Scheer, A., 1992. Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). Inst. fur Wirtschaftsinformatik.

Knottenbelt, J., Clark, K., 2004. An architecture for contract-based communicating agents. In: Proceedings of the 2nd European Workshop on Multi-Agent Systems.
URL http://www.doc.ic.ac.uk/ jak97/Papers/jakklc04.pdf

Kumar, A., Yao, W., Nov. 2009. Process Materialization using Templates and Rules to Design Flexible Process Models. In: Governatori, G., Hall, J., Paschke, A. (Eds.), Rule Interchange and Applications (RuleML 2009). Las Vegas, pp. 122–136.

Lu, R., Sadiq, S., 2007. A survey of comparative business process modeling approaches. In: Business Information Systems. Springer, pp. 82–94.

Lu, R., Sadiq, S., Governatori, G., 2009. On managing business processes variants. Data & Knowledge Engineering 68 (7), 642–664.

Lu, R., Sadiq, S., Padmanabhan, V., Governatori, G., 2006. Using a temporal constraint network for business process execution. In: Proceedings of the 17th Australasian Database Conference-Volume 49. Australian Computer Society, Inc., pp. 157–166.

Madhusudan, T., Zhao, J., Marshall, B., 2004. A case-based reasoning framework for workflow model management. Data & Knowledge Engineering 50 (1), 87–115.

Maghraoui, K., Meghranjani, A., Eilam, T., Kalantar, M., Konstantinou, A., 2006. Model driven provisioning: Bridging the gap between declarative object models and procedural provisioning tools. Middleware 2006, 404–423.

Mansar, S., Reijers, H. A., 2005. Best practices in business process redesign: validation of a redesign framework. Computers in Industry 56 (5), 457–471.

Mar´ın, R. H., Sartor, G., 1999. Time and norms: a formalisation in the event-calculus. In: ICAIL '99: Proceedings of the 7th international conference on Artificial intelligence and law. ACM Press, New York, NY, USA, pp. 90–99.

Mendling, J., Reijers, H., van der Aalst, W., 2010. Seven process modeling guidelines (7pmg). Information and Software Technology 52 (2), 127–136.

Muehlen, M., Recker, J., 2008. How much language is enough? theoretical and practical use of the business process modeling notation. In: Advanced Information Systems Engineering. Springer, pp. 465–479.

Nau, D., Ghallab, M., Traverso, P., 2004. Automated Planning: Theory & Practice. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Nelson, M., Peterson, J., Rariden, R., Sen, R., 2009. Transitioning to a business rule management service model: Case studies from the property and casualty insurance industry. Information & Management.

Neumann, L., 2009. Exploring The Perceived Business Value Of The Flexibility Enabled By Information Technology Infrastructure. Information & Management 46 (2). Object Management Group, Februari 2006. Business Process Modeling Notation (BPMN). OMG Document – dtc/06-02-01.

O'Conor, M., July 2005. The implications of Sarbanes-Oxley for non-US IT departments. Network Security 2005 (7), 17–20.

Oesterreichische Nationalbank (OeNB), 2006. Guidelines on credit risk management – credit approval process and credit risk management. http://www.oenb.at.

OMG, 2004. Uml 2.0 superstructure specification.

OMG, 2006. Business process modeling notation (bpmn) 1.1.

Paschke, A., Bichler, M., 2005. SLA Representation, Management and Enforcement. In: EEE '05: Proceedings of the 2005 IEEE International Conference on e-Technology, eCommerce and e-Service (EEE'05). IEEE Computer Society, Washington, DC, USA, pp. 158–163.

Peltz, C., 2003. Web services orchestration and choreography. IEEE Computer 28 (10), 46–52.

Pesic, M., 2008. Constraint-based workflow management systems: Shifting control to users. Ph.D. thesis, Eindhoven University of Technology.

Pesic, M., Schonenberg, H., van der Aalst, W. M. P., 2007a. Declare: Full support for loosely-structured processes. In: Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007). IEEE Computer Society, pp. 287–300.

Pesic, M., Schonenberg, M. H., Sidorova, N., van der Aalst, W. M. P., 2007b. Constraintbased workflow models: Change made easy. In: Meersman, R., Tari, Z. (Eds.), OTM Conferences (1). Vol. 4803 of Lecture Notes in Computer Science. Springer, pp. 77–94.

Pesic, M., van der Aalst, W., 2006a. A declarative approach for flexible business processes management. In: Business Process Management Workshops. Springer, pp. 169–180.

Pesic, M., van der Aalst, W., 2006b. A declarative approach for flexible business processes management. In: Business Process Management Workshops. Springer, pp. 169–180.

Pesic, M., van der Aalst, W. M. P., 2006. A declarative approach for flexible business processes management. In: Business Process Management Workshops. pp. 169–180.

Porter, M., 1985. Competitive Advantage. Free Press, New York.

R-Moreno, M., Kearney, P., Meziat, D., 2000. A case study: using workflow and ai planners. In: 19th Workshop of the UK Planning and Scheduling (PLANSIG2000), Milton Keynes (UK).

Recker, J., 2010. Opportunities and constraints: the current struggle with BPMN. Business Process Management Journal 16 (1), 181–201.

Recker, J., Indulska, M., Rosemann, M., Green, P., 2005. Do process modelling techniques get better? A comparative ontological analysis of BPMN. In: Proceedings of the 16th Australasian Conference on Information Systems. Australasian Chapter of the Association for Information Systems, Sydney, Australia.

Recker, J., Rosemann, M., Indulska, M., Green, P., 2009. Business process modeling: a comparative analysis. Journal of the Association for Information Systems 10 (4), 333–363.

Regev, G., Soffer, P., Schmidt, R., 2006. Taxonomy of flexibility in business processes. In: Regev, G., Soffer, P., Schmidt, R. (Eds.), BPMDS.

Reichert, M., Dadam, P., 1998. ADEPTflex-supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems 10 (2), 93–129.

Reijers, H., Mans, R., van der Toorn, R., 2009. Improved model management with aggregated business process models. Data & Knowledge Engineering 68 (2), 221 – 243.

Rhee, S., Cho, N., Bae, H., 2010. Increasing the efficiency of business processes using a theory of constraints. Information Systems Frontiers 12 (4), 443–455.

Rolstad˚as, A., 1995. Business process modeling and reengineering. Performance Management: A Business Process Benchmarking Approach, 148–150.

Ross, R. G., 2003. Principles of the Business Rule Approach. Addison-Wesley Professional.

Rychkova, I., Regev, G., Wegmann, A., 2008. Using declarative specifications in business process design. International Journal of Computer Science and Applications 5 (3b), 45–68.

Sadiq, S., Indulska, M., 2011. The compliance enabled enterprise-process is the product. Compliance & Regulatory Journal 2008 (5), 27–31.

Sadiq, S. W., Orlowska, M. E., Sadiq, W., 2005. Specification and validation of process constraints for flexible workflows. Information Systems 30 (5), 349–378.

Schmidt, K., Simone, C., 1996. Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. Computer Supported Cooperative Work 5 (2/3), 155–200.

Schmidt, K., Simonee, C., 1996. Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. Computer Supported Cooperative Work (CSCW) 5 (2), 155–200.

Schonenberg, H., Mans, R., Russell, N., Mulyar, N., Aalst, W., 2008a. Process flexibility: A survey of contemporary approaches. Advances in Enterprise Engineering I, 16–30.

Schonenberg, H., Mans, R., Russell, N., Mulyar, N., Van der Aalst, W., 2008b. Towards a taxonomy of process flexibility. In: Proceedings of the CAiSE. Vol. 8. pp. 81–84.

Schonenberg, H., Weber, B., Dongen, B., Aalst, W., 2008c. Supporting Flexible Processes through Recommendations Based on History. In: Proceedings of the 6th International Conference on Business Process Management. Springer-Verlag, p. 66.

Sinur, J., 2009. The art of science of rules vs process flows. Tech. Rep. G00166408, Gartner.

Suchman, L. A., 1987. Plans and situated actions: the problem of human-machine communication. Cambridge University Press, New York, NY, USA.

Suchman, L. A., 1995. Making work visible. Communications of the ACM 38 (9), 56–64.

Swenson, K., Palmer, N., 2010. Mastering the Unpredictable: How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done. Meghan-Kiffer Press.

Van der Aalst, W., Adams, M., Ter Hofstede, A., Pesic, M., Schonenberg, H., 2009.

Flexibility as a service. In: Database Systems for Advanced Applications. Springer, pp. 319–333.

Van Der Aalst, W., et al., 1998. Three good reasons for using a Petri-net-based workflow management system. Kluwer international series in engineering and computer science, 161–182. van der Aalst, W. M. P., Jablonski, S., September 2000. Dealing with workflow change: identification of issues and solutions. International Journal of Computer Systems Science and Engineering 15 (5), 267–276.

van der Aalst, W. M. P., van Hee, K. M., Houben, G. J., 1994. Modelling and analysing workflow using a Petri-net based approach. In: De Michelis, G., Ellis, C., Memmi, G. (Eds.), Proceedings of the 2nd Workshop on Computer-Supported Cooperative Work, Petri nets and related formalisms. pp. 31–50. van der Aalst, W. M. P., Weske, M., Gru¨nbauer, D., 2005. Case handling: a new paradigm for business process support. Data & Knowledge Engineering 53 (2), 129–162.

Wagner, G., 2003. The agent-object-relationship metamodel: towards a unified view of state and behavior. Information Systems 28 (5), 475–504.

Wang, J., Kumar, A., 2005. A framework for document-driven workflow systems. Business Process Management, 285–301.

Wang, M., Wang, H., 2006. From process logic to business logic – A cognitive approach to business process management. Information & Management 43 (2), 179–193.

Weber, B., Reichert, M., Rinderle-Ma, S., 2008. Change patterns and change support features - enhancing flexibility in process-aware information systems. Data & Knowledge Engineering 66 (3), 438 – 466.

Weber, B., Reijers, H., Zugal, S., Wild, W., 2009a. The declarative approach to business process execution: An empirical test. In: Proceedings of the 21st International Conference on Advanced Information Systems Engineering. Springer, pp. 470–485.

Weber, B., Sadiq, S., Reichert, M., 2009b. Beyond rigidity–dynamic process lifecycle support. Computer Science-Research and Development 23 (2), 47–65.

Weske, M., van der Aalst, W., Verbeek, H., 2004. Advances in business process management. Data & Knowledge Engineering 50 (1), 1 – 8.

Woolridge, M., Wooldridge, M. J., 2001. Introduction to Multiagent Systems. John Wiley & Sons, Inc., New York, NY, USA.

Xiao, L., Zheng, L., 2010. Business process design: Process comparison and integration. Information Systems Frontiers, 1–12.

Yang, G., 2004. Process library. Data & Knowledge Engineering 50 (1), 35 – 62.

Yolum, P., 2005. Towards design tools for protocol development. In: AAMAS ’05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems. pp. 99–105.

Zisman, M., 1977. Representation, specification and automation of office procedures. Ph.D. thesis, Wharton School.

zur Muehlen, M., Indulska, M., 2010. Modeling languages for business processes and business rules: A representational analysis. Information Systems 35 (4), 379–390.