Felix Kossak · Christa Illibauer
Verena Geist · Christine Natschläger
Thomas Ziebermayr
Bernhard Freudenthaler
Theodorich Kopetzky
Klaus-Dieter Schewe

Hagenberg Business Process Modelling Method

Springer

Hagenberg Business Process Modelling Method

Felix Kossak · Christa Illibauer
Verena Geist · Christine Natschläger
Thomas Ziebermayr · Bernhard Freudenthaler
Theodorich Kopetzky · Klaus-Dieter Schewe

Hagenberg Business Process Modelling Method

Springer

Felix Kossak
Christa Illibauer
Verena Geist
Christine Natschläger
Thomas Ziebermayr
Bernhard Freudenthaler
Theodorich Kopetzky
Klaus-Dieter Schewe

Software Competence Center Hagenberg GmbH
Hagenberg im Mühlkreis
Austria

ISBN 978-3-319-30495-3 ISBN 978-3-319-30496-0 (eBook)
DOI 10.1007/978-3-319-30496-0

Library of Congress Control Number: 2016934680

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

Preface

The whole is greater than the sum of its parts

—Aristotle

The Hagenberg Business Process Modelling (H-BPM) method constitutes a proposal for the design of Business Process Management (BPM) systems, which in addition to process modelling comprises several other important aspects like actor modelling, user interaction modelling, and an enhanced communication concept. On top of these aspects, we propose the enhanced Process Platform (eP^2) architecture to integrate the different models in a single tool.

The presented book gives insight into major results of a fundamental research project on business process modelling, called Vertical Model Integration (VMI), and its successor project, VMI 4.0, performed at the Software Competence Center Hagenberg (SCCH), Austria. It builds on a previous book, *A Rigorous Semantics for BPMN 2.0 Process Diagrams*[1], and like the latter, draws from our experience in large-scale business software development projects where we have experienced the need to go beyond BPM languages like BPMN.

This book mainly addresses researchers in the area of business process modelling, although we hope that it may also provide useful input to developers of modelling tools.

The introduced method is named after Hagenberg, a village in the district of Freistadt in the state of Upper Austria, situated on one of the green hills on the southern edge of the Bohemian Massif. Hagenberg is well known for the *Softwarepark Hagenberg*, a technology centre comprising 12 research institutes, 23 academic study programmes, and about 70 companies in the IT domain.

[1] Kossak, F., Illibauer, C., Geist, V., Kubovy, J., Natschläger, C., Ziebermayr, T., Kopetzky, T., Freudenthaler, B., Schewe, K.D.: A Rigorous Semantics for BPMN 2.0 Process Diagrams. Springer (2015)

The SCCH, where the H-BPM method has been developed, is one of the largest research institutions in the Softwarepark Hagenberg.

The research reported in this monograph has been partly supported by the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center SCCH.

This publication has been written within the projects “Vertical Model Integration (VMI)” and “VMI 4.0”. Both projects have been supported within the programme “Regionale Wettbewerbsfähigkeit OÖ 2007–2013” by the European Fund for Regional Development as well as the State of Upper Austria.

This work was also supported in part by the *Austrian Research Promotion Agency* (FFG) under grant no. 842437.

Software Competence Center Hagenberg GmbH,
Hagenberg im Mühlkreis, Austria

Felix Kossak
Christa Illibauer
Verena Geist
Christine Natschläger
Thomas Ziebermayr
Bernhard Freudenthaler
Theodorich Kopetzky
Klaus-Dieter Schewe

Contents

Acronyms

ADL	Architectural Description Language
ARIS	Architecture of Integrated Information Systems
ASM	Abstract State Machine
AST	Abstract Syntax Tree
AUVA	Austrian Social Insurance Company for Occupational Risks
BPEL	Business Process Execution Language
BPM	Business Process Management
BPML	Business Process Modelling Language
BPMN	Business Process Model and Notation
CCS	Calculus of Communicating Systems
CPN	Coloured Petri-Net
CPS	Cyber Physical Systems
CPPS	Cyber Physical Production Systems
CSP	Communicating Sequential Process
DL	Description Logic
DSL	Domain-Specific Language
EAI	Enterprise Application Integration
eP^2	Enhanced Process Platform
EPC	Event-Driven Process Chain
ER	Entity-Relationship
H-BPM	Hagenberg Business Process Modelling
HCI	Human–Computer Interaction
HDM	Hypertext Design Model
ISO	International Organization for Standardization
IT	Information Technology
JSF	JavaServer Faces
MB-UIDE	Model-Based User Interface Development Environment
MDA	Model-Driven Architecture
OCL	Object Constraint Language
OMG	Object Management Group
OOHDM	Object-Oriented Hypertext Design Model

R2ML	REWERSE Rule Markup Language
RMM	Relationship Management Methodology
S-BPM	Subject-Oriented Business Process Management
SBVR	Semantics of Business Vocabulary and Rules
SME	Small- and Medium-sized Enterprise
SQL	Structured Query Language
SysML	Systems Modelling Language
UIML	User Interface Markup Language
UML	Unified Modeling Language
WebML	Web Modeling Language
WS-BPEL	Web Services Business Process Execution Language
XML	Extensible Markup Language
YAWL	Yet Another Workflow Language

Chapter 1
Introduction

This book contains a proposal for designing Business Process Management (BPM) systems which comprise much more than just process modelling. We see process modelling as a possible first step in BPM (though one could start with user-centric modelling as well, see [12]); but when it comes to business process *execution*, much more is needed for full-scale software support. At this point, if not already before, users (workers) will have to be added to the model, with issues from access control to user interfaces, and business data as well as cross-platform communication concepts have to be integrated.

We forgo an introduction to BPM in this place, as we rely on readers already having experience and knowledge in this field. Comprehensive introductions focussing on a classical approach can be found, amongst others, in the book of Dumas et al. on *Fundamentals of Business Process Management* [11] and in the book of ter Hofstede et al. on *Modern Business Process Automation* [45]; the book of Fleischmann et al. on *Subject-Oriented Business Process Management* [12], for instance, presents an alternative approach.

This book builds on a previous book, *A Rigorous Semantics for BPMN 2.0 Process Diagrams* [20] (not necessarily required for comprehending the current book), in which a formal specification of a purified semantics for the Business Process Model and Notation (BPMN) 2.0 was used, amongst others, to discuss a range of shortcomings of the BPMN standard [29]. Still, we think that BPMN, as a well-received international standard with ample tool support, forms a good basis for further development to service all the needs of BPM in practice.

Based on such a purified BPMN variant, we now present proposals for several important issues in BPM which have not or hardly been regarded in the BPMN 2.0 standard. Due to significant changes in comparison with BPMN as it stands today and, most of all, the extensive supplements we now propose, we have chosen to give the result an own name, the *Hagenberg Business Process Modelling (H-BPM)* method, named after the Upper Austrian village where it has been designed. The main issues discussed here are actor and user interaction modelling, rounded off by an enhanced communication concept. On top of these, we propose an enhanced

F. Kossak et al., *Hagenberg Business Process Modelling Method*,
DOI 10.1007/978-3-319-30496-0_1

Process Platform (eP^2) architecture capable of integrating all these aspects (as well as data modelling) in a single tool. We also describe how the different aspects and models work together.

In order to render models executable, the semantics of the modelling language needs to be described rigorously enough to prevent deviating interpretations by different tools. The semantics of the necessary concepts introduced in this book are defined using the Abstract State Machine (ASM) method [7], which is a system engineering method for developing software and systems seamlessly from requirements capture to their implementation. Within a precise but simple conceptual framework, the ASM method allows a modelling technique which integrates dynamic (operational) and static (declarative) descriptions, and an analysis technique that combines verification and validation methods at any desired level of detail. ASMs are an extension of finite state machines. The method has a rigorous mathematical foundation, yet a practitioner needs no special training to use the method since ASMs can be correctly understood as virtual machines working over abstract data structures.

1.1 Motivation

Today the success of enterprises and organisations depends very much on the speed with which they can create new business processes and adapt existing ones to react to increasingly fast changes in the environment and to take advantage of new trends and events.

Today's Trends—Factors of Influence

Due to the fact that communication is still becoming increasingly faster, easier, and more intense, more and more participants are joining the global business landscape to find and grab new opportunities [21]. Not only large, international organisations, but also Small and Medium-sized Enterprises (SMEs) now have to adapt their business to face up to global competition.

Factors in pushing the need for more flexibility of workflows but also for more comprehensive and better integrated workflows are trends like just-in-time delivery, outsourcing and offshoring, product or service customisation right down to individualised products, and requirements of traceability not only of products themselves but also of raw materials and up to disposal or recycling. Extreme customisation as well as high integration of workflows transcending single places and organisations is envisaged, for instance, by the European "Industrie 4.0" initiative.

The term "Industrie 4.0", initialised by the German government [8], is designed to signify a fourth industrial revolution. The first industrial revolution (at the end of the eighteenth century) was fostered by the introduction of mechanical production facilities with the aid of water and steam power. The second industrial revolution (at the beginning of the twentieth century) was triggered by mass production with the aid of electricity. In the 1970s, the third industrial revolution was caused by the emergence of electronics and Information Technology (IT) [15].

Industrie 4.0 applies new trends from the information and communication technology to production systems. The goal is to create intelligent machines, logistic systems, and equipment that independently communicate with each other, that are able to trigger suitable events, and that are even able to mutually control each other [13]. Such networked (mostly over the internet) and communicating systems are called Cyber Physical Systems (CPS) or, when they are used in production, Cyber Physical Production Systems (CPPS) [26]. When, in addition to the production system, also sourcing and delivery (i.e. the supply chain) are included, such factories are called "Smart Factories". Special, domain-specific applications include "Smart Grid", "Smart Buildings", "Smart Products", "Smart Logistics", and "Smart Mobility"; all of them entail the same issues regarding communication between IT and software technologies on the one hand and electronic parts on the other [13].

With regard to implementations of Industrie 4.0 projects, Kagerman [15] emphasise the importance of a methodical approach comprising every aspect from requirements to product architecture and manufacture of the product. Furthermore they mention modelling and integration (horizontal integration through value networks, integration of engineering across the value chain, as well as vertical integration and networked manufacturing systems) as an essential challenge of enterprises for being prepared for Industrie 4.0.

H-BPM—A Holistic Modelling Approach

We have designed H-BPM to cover a range of aspects which we think are necessary to model business processes at the level required for automation and fit for Industrie 4.0. To motivate the holistic approach, for a start, consider an analogy from grammar: a sentence must—at least—consist of the central parts as are subject, predicate, and often also object in order to be valid and understandable; similarly, a business process with all its different aspects can only be understood in a holistic and intelligible way if the applied modelling method covers precise definitions for the essence of all BPM constructs and, particularly, supports accurate integration. To draw an analogy between a sentence and an H-BPM diagram, (i) the predicate is represented by a basic control-flow view on the process, including activities with deontic classifications to describe modalities and an enhanced communication concept for serving more sophisticated communication patterns for business processes, (ii) the subject is addressed by an extended actor modelling approach that provides a task-based assignment for actors, and (iii) the object is given by process data which corresponds to typed nodes of dialogues for user tasks. Thus, H-BPM is able to support the equivalent of full sentences. For only if all aspects are considered in a formal and integrated way, business processes can be understood with their full contexts.

Traditional process modelling languages like BPMN provide good support for the control-flow perspective and medium support for the data perspective, but the resource perspective is not well supported [49]. In [28], the limited support for actor modelling provided by rigid swimlane concepts is discussed in detail. Missing integration mechanisms for user interaction and data modelling are furthermore likely to pose communication problems and inconsistencies when planning and developing process-oriented systems [3]. Also simple communication patterns like those provided in BPMN often do not suffice for modelling the interplay between different processes, especially between very heterogeneous systems [19]. In particular, more flexibility and customisation are needed when human actors and user interaction are considered.

The number of models required for those different aspects, such as functionality, actors, user interaction, and communication, makes modelling information systems a complex endeavour [41]. In addition, those models are generally related to different abstraction layers, leading to discrepancies when integrating them.

The motivation for the development of a comprehensive modelling method proposing a design for BPM systems which comprise much more than just process modelling stems from several of our industrial projects (see [10, 28]), where business analysts and software developers struggled with redundancies and inconsistencies in system documentation due to missing integration. While different views on business processes represented by different models have been described before, to our knowledge, these different models have never been brought together in the available literature. In fact, it is their interplay which makes them useful in practice, and this interplay is not trivial.

Therefore, we suggest the H-BPM method for seamless modelling of business processes. Thereby, we take a static view regarding software component integration as well as a dynamic, runtime-related view on model integration. As a result, this new modelling method is able to seamlessly integrate different aspects of business process modelling, including organisational (actor), user interaction, data, and communication models, on all levels of abstraction.

Motivation for Formal Specification

Meeting modern challenges and the requirements for Industrie 4.0 will require trans-organisational workflows and a high level of interconnection between different systems. Consequently, it must be possible to exchange business process models between different organisations without undesired loss of information. Exchange of models, in turn, requires a rigidly specified standard.

The BPMN standard, arguably the most important international standard for business process modelling, is formulated in natural language. It is not so surprising, then, that we have identified numerous ambiguities, gaps, and inconsistencies in this standard [20]. Consequently, model interchange between different tools, despite being supported by an XML-based exchange format defined in the standard, is limited, as experiments by us, one of our partners, and others have shown; typically, imported models can be displayed but not run. Also the semantics of certain elements, e.g. the inclusive gateway, are interpreted differently, or such elements are not supported at all.

Thus, it is particularly important to guarantee that the executable behaviour of a process model is exactly as intended by the process designer. Consequently, the semantics of the modelling language and method needs to be specified *formally* such that no room for interpretation is left. Furthermore, a formal model allows for checking for certain properties, including safety and liveness properties.

We chose the notation of ASM for its closeness to natural language and its consequent understandability (despite its rigour). We can thereby also seamlessly build on our work in [20], where we rigorously defined a semantics for process models, following the BPMN standard as closely as possible; see also our discussion of the topic there.

1.2 Related Work and Comparison with Related Methods

In this section, an overview of related BPM methods and languages and a comparison with respect to considered aspects and their overall integration is provided. The results are summarised in Table 1.1.

The Subject-Oriented Business Process Management (S-BPM) method puts the focus on the subject of a process and defines two views: (i) the communication view, which depicts the process in its entirety (how the subjects collude in the process and which messages they interchange) and (ii) the internal subject view, which depicts the internal behaviour of a subject (which actions are to be processed and how to react or trigger communication). The formal foundation of S-BPM [32] is based on the Calculus of Communicating Systems (CCS) introduced by Milner [25]. The integration is done during process design; first the communication structure is specified and subsequently each subject models its behaviour from his or her perspective [22, 23].

Architecture of Integrated Information Systems (ARIS) is a method for optimising business processes as well as for implementing application systems. ARIS defines five views, which are symbolically presented in the form of the so-called ARIS house: (i) the organisational view (describing the organisational structure), (ii) the data view (business data and information), (iii) the control- or process-view (behaviour processes and their relations to services, organisation, functions and data), (iv) the functional view (tasks and business objectives, function hierarchies, etc.), and (v) the product (and service) view (products and services, their structures, relations, and product/service trees) [40]. ARIS further provides integration of concepts from other views [42, 48]. Formerly, the main ARIS model for processes was based on Event-Driven Process Chains (EPCs) [9], but recently also support for BPMN [2] has been added.

ADONIS is a business process management tool for designing, documenting, and optimising business processes, which supports business process management systems and BPMN 2.0. The idea of ADONIS is to encompass the phases identified within the business process management framework with the theory of a permanent lifecycle as depicted in most process management systems. The four key elements considered within ADONIS are: (i) products/services, (ii) processes, (iii) organisational structures, and (iv) information theory (including their dependencies) [5, 6].

The BPM-D framework is an architecture and toolset for establishing the BPM discipline in an organisation. The four major components of the framework are: (i) BPM-D Process: The process model is a reference model for project-based and asset-based processes and is detailed through four layers of decomposition with descriptions, reference methods, and best-practice examples. (ii) BPM-D Data: The information model is a reference model that details all major data entities. (iii) BPM-D Organisation: The organisation model is a reference model that comprises internal and external roles, their responsibilities and key performance metrics. It further includes a reference organisation structure together with examples of how this has been implemented in other organisations. (iv) BPM-D Value: The value model details the potential areas where value can be found and outlines pragmatic approaches to focus all development efforts on delivering this value [17].

The Horus method covers the whole life cycle of business process engineering and suggests steps to extend a process model with additional elements and to link (integrate) these elements with each other. The holistic business process management considers the following aspects: (i) process modelling with Petri nets, (ii) object modelling with business objects (e.g., documents, data objects, etc.), and (iii) organisational modelling. Besides the process-view, the Horus method also takes the service- and business-view with aspects such as ratio and risk analyses into account [43].

BPMN is a graphical modelling language for business processes and an international standard issued by the Object Management Group (OMG), a well-established group with a strong foundation in the industry [29]. It was formally published by the International Organization for Standardization (ISO) as the 2013 edition standard ISO/IEC 19510 [14]. BPMN has been widely adopted and is supported by various tools. Nevertheless, BPMN has major drawbacks like the lack of integrated user interaction and data modelling, a restricted support on organisational modelling and communication as well as only implicit expression of modalities [20].

The Yet Another Workflow Language (YAWL) system [45] is a service-oriented architecture and consists of an extensible set of YAWL services, each offering one or more interfaces. YAWL supports three different perspectives: control-flow, data, and resources. The formal foundation of YAWL makes its specifications unambiguous and enables automated verification. Additionally, YAWL offers two verification approaches, one based on Reset nets (Petri nets with reset arcs) and another one based on transition invariants. Aldred defines "process integration patterns", which could be seen as a support for communication aspects in an extension [1].

EPC is a popular business process modelling language introduced by Keller et al. [16] and defines the sequence-related connection of functions which are triggered by events. The main node types of EPCs are activities (functions), events and gateways (connectors). Connector operations describe "how" elements are connected (e.g. conjunction, disjunction or adjunction (and/or)) whereas the connection type defines which elements of the models are connected. An extension of EPC with data and resources is called eEPC [24, 39, 47].

The Unified Modeling Language (UML) [31] is a general-purpose modelling language maintained by the OMG and mainly used in software engineering. UML 2.5 defines 14 types of diagrams divided into the categories of structure and behaviour diagrams. Classified as structure diagrams are class, component, object, composite structure, deployment, package, and profile diagrams. The category of behaviour diagrams includes activity diagrams, use case diagrams, state machine diagrams and the sub-class of interaction diagrams comprising sequence diagrams, interaction overview diagrams, communication diagrams, and timing diagrams [31]. However, integration between these types of diagrams is only partly given based on the corresponding UML metamodel. Furthermore, also the integration between UML and BPMN diagrams, both specified by the OMG, is limited (cf. [18]). Additionally, for the design of user interfaces, different UML profiles have been specified [4, 33, 37]. Silva et al. provide a case study comprising common modelling problems of UML for modelling user interfaces [36].

The Systems Modelling Language (SysML), first released in 2006 by OMG, is a general-purpose visual modelling language for systems engineering. The systems may include hardware, software, information, processes, personnel, and facilities. SysML reuses a subset of UML 2 and provides additional extensions. SysML is especially applied to specify requirements, structure, behaviour, allocations, and constraints on system properties to support engineering analysis. SysML may be used to create models of the system (the entire model) as well as for viewpoints or view models of different stakeholders [30].

Petri nets, introduced by Adam Petri [34, 35], are a graphical and mathematical modelling language for concurrent, asynchronous, and distributed systems. A Petri net is a directed, weighted, bipartite graph, consisting of two kinds of nodes: (i) “places” (depicted as circles) and (ii) “transitions” (bars or boxes). Connecting arcs (either from a transition to a place or vice versa) can be labelled with weights. A marking can assign a non-negative integer k to a place (i.e. the place is marked with k “tokens”) [27]. In modelling, places represent conditions whereas transitions represent events. A change of a state is denoted by the movement of tokens from places to places, effected by the firing of a transition. Van der Aalst studies the inability of classical Petri nets to model data and time and refers to three important extensions of Petri nets (called high-level Petri nets): (i) extension with colour to attach data value to tokens, (ii) extension with time, and (iii) extension with hierarchy for structuring large models [44]. Amongst others, Petri nets are used to formally specify the semantics of process modelling languages, e.g. YAWL (see above).

Workflow nets (WF-nets), introduced by van der Aalst [44], constitute a sub-type of Petri nets to model workflow process definitions. In a WF-net, tasks are modelled by transitions, conditions are modelled by places, and cases are modelled by tokens. WF-nets further satisfy the following requirements: they have exactly one input and one output place and there are no dangling tasks and/or conditions, so every transition is located on a path from the input to the output place [44, 46].

Like the proposed H-BPM method, S-BPM considers the business process model, actors, and the communication aspect, and even provides displaying user interfaces by triggering services. Similarly, ADONIS takes processes and organisational structures into account. ARIS suggests five views including processes, actors, and data. BPM-D comprises a process, actor, and data model as well as a value model to define the potential areas where value can be found. In addition, the Horus method considers not only the process-view with the process, data and organisational model, but also the business- and service-view throughout the whole business process lifecycle. All

Table 1.1 Comparison of BPM methods and languages

Name	Process	Organsational	Modality	User interaction	Data	Communication	Formal basis	Integration
BPM methods								
H-BPM	Yes	Yes	Yes	Yes	Partly	Yes	Yes	Yes
S-BPM	Yes	Yes	No	Service[a]	No	Yes	Yes	Yes
ARIS	Yes	Yes	No	Partly	Yes	No	Semi	Yes
ADONIS	Yes	Yes	No	No	No	No	No	No
BPM-D	Yes	Yes	No	No	Yes	No	No	No
Horus	Yes	Yes	No	No	Yes	No	Yes	Partly
BPM languages								
BPMN	Yes	Partly	No	No	Partly	Partly	Semi	Partly
YAWL	Yes	Yes	No	Yes	Yes	Ext[b]	Yes	Yes
EPC	Yes	Ext[b]	No	No	Ext[b]	Ext[b]	Semi	Ext[b]
UML/	Yes	Partly	No	Profile[c]	Yes	Partly	Semi	Partly
SysML	Yes	Yes	No	No	Yes	Partly	Semi	Partly
Petri net/	Yes	No	No	No	No	No	Yes	No
WF-net	Yes	Ext[b]	No	No	Ext[b]	No	Yes	Ext[b]

[a] By calling services
[b] In an extension
[c] In specific profiles

investigated approaches except ADONIS and BPM-D are built on a formal (or at least semi-formal) foundation. However, none of the aforementioned methods considers modalities and only S-BPM and ARIS (rudimentarily) define Human–Computer Interaction (HCI) as we provide it in our approach to user interaction modelling. Model integration, a main focus of the H-BPM method, is provided by S-BPM, ARIS, and partly by the Horus method.

Considering related BPM languages, BPMN provides only restricted support for data and organisational modelling and lacks integration of these aspects. Furthermore, UML, SysML, and YAWL consider data modelling and YAWL as well as SysML additionally support organisational aspects. All other investigated modelling languages either do not support organisational and data aspects or facilitate them either partly or only in an extension to the language (see Table 1.1). Only YAWL and UML (in specific profiles) support user interactions. Model integration is provided by YAWL and in an extension by EPC and WF-nets, however, partial integration of UML-based models (and also SysML models) can be done based on the corresponding UML metamodel [38, p. 187].

1.3 Outline

This book is structured as follows. We start with the way in which a user who has to perform tasks should be confronted with a process model. First, such a user need not be confronted with the whole process diagram (though knowledge about the whole process may often be very helpful); what they need in the first place is to be presented with a list of tasks which they are supposed to perform at a given moment. Second, those tasks may have different status—some tasks *have* to be performed (are obligatory), while others are optional (may but need not be performed); yet other tasks can be alternatives with respect to a group of further tasks. All these *modalities* can depend on certain conditions, and some tasks may even be forbidden under certain circumstances.

In a classical process diagram, the modality of a task is implicitly given by the structure of the whole diagram (or a part of the diagram). But this is not a comfortable and reliable way of detecting whether one must or may or must not perform a certain task. Instead, such modalities—like obligation or optionality—can be described in different flavours of deontic logic, which is the subject of Chap. 2. Even beyond looking at single tasks, the introduction of deontic modalities can often simplify process diagrams and render them better understandable. We use colour highlighting in combination with abbreviations for deontic operators to differentiate between modalities within a process diagram, and show how deontically modified diagrams are related to classical BPMN-style process diagrams.

Actor modelling, introduced in Chap. 3, then builds upon this deontic business process diagrams and allocates different modalities to different actors, depending on role permissions. We present a layered actor modelling approach with different views on the involvement of actors and their roles in a particular process, including task-based assignment of users, a hierarchical role model, and rules to define dependencies between tasks. Those rules can also be checked for consistency and derived rules can be generated.

Chapter 4 focuses on user interaction modelling, and thus on the user interface for those people who actually have to perform the user tasks defined in a process diagram. Such users, in their daily work, have a totally different view on a process than a business process analyst or higher level manager (compare, for instance, [12]). The core of a worker's view is represented by a *worklist* of tasks from which they can select. Selecting an item in their worklist, users are led through dialogues consisting of reports and forms. They give access to the data needed for the task and allow to perform actions (like entering data). Workflow charts also come with an own workflow model which, amongst others, allows to further structure tasks. Thereby, a user can be led through a dynamically assembled succession of dialogues, depending, amongst others, on their own actions and decisions.

In Chap. 5, we discuss how the event concept of BPMN can be generalised. BPMN provides a set of specific event types, like "Message" or "Signal", but those types do not cover all possibilities for communication. Those types are distinguished by different properties, and together with different types of *event pools*, we exploit those properties to define a generic event concept. Amongst others, event pools allow users to select messages and to subscribe or unsubscribe to public event pools, i.e. notification sources.

Having proposed several extensions to the workflow-centric language of BPMN and its purified version according to [20], we deal with the question of model integration in Chap. 6. We show how the different part models fit together, using a simple example process. We introduce the enhanced Process Platform (eP^2) architecture which binds all the different components together at runtime, so that the various business process modelling aspects can be supported by a single tool (cf. Sect. 6.4).

A detailed specification of the (eP^2) architecture and the integration of the different components therein are then given in Chap. 7.

Still more can be done towards a unified, comprehensive approach to BPM, though. In the Outlook in Chap. 8, we briefly recap the components of the H-BPM model introduced in this book and point out need for future work.

1.4 Recommendations for Readers

We tried to make the specific chapters as independent from each other as possible. However, *Actor Modelling* (Chap. 3) builds heavily on deontic process diagrams (Chap. 2), thus we suggest to read those two chapters together. Chapter 2 can be understood independently, however.

Not at all independent are the chapters on *Horizontal Model Integration* (Chap. 6) and the specification of the eP^2 architecture (Chap. 7), because they show how the separate parts presented in the previous chapters all fit together. Chapter 6 gives an overview of the eP^2 architecture (in Sect. 6.4) and can thus be understood independently of Chap. 7, but Chap. 7 builds on Chap. 6.

With those exceptions, it should be possible to read the more specific chapters independently of each other.

As in [20], wherever we consider it necessary to define the semantics of a concept rigorously, we resort to the ASM method. For our purposes in this book, ASM can simply be seen as “a rather intuitive form of abstract pseudo-code” [7, p. 2]. The foundation of the ASM method is a rigorous mathematical theory based on automata whose states are defined by arbitrarily complex data structures.

Functions define the data structure of an ASM. Concrete values of the parameters of a function define a *location* (comparable to a “memory address” of a computer at runtime), and concrete values of the functions for all locations define a particular state of the automaton. *Rules* define state transitions by modifying the function values at a finite number of locations. *Derived functions*, which constitute an important auxiliary element in ASM models, compute values from a combination of “proper” functions (or data) at runtime.

We trust that the ASM functions and rules we define in this book are intuitive to understand without any background knowledge of ASMs. For an understanding of the semantic subtleties, we recommend to consult “the ASM Book” by Börger and Stärk [7]; for a shortish introduction, see also [20, Chap. 3], amongst many other sources.

References

1. Aldred, L.: Process integration. In: ter Hofstede, A.M., van der Aalst, W.M.P., Adams, M., Russell, N. (eds.) Modern Business Process Automation: YAWL and its Support Environment, pp. 489–511. Springer, Heidelberg (2010)
2. ARIS Community: BPMN process modeling & free modeling tool. http://www.ariscommunity.com/aris-express/bpmn-2-free-process-modeling-tool (2015). Accessed 18 Sept 2015
3. Auer, D., Geist, V., Draheim, D.: Extending BPMN with submit/response-style user interaction modeling. In: Proceedings of CEC’09, pp. 368–374. IEEE Computer Society (2009)
4. Blankenhorn, K., Jeckle, M.: A UML profile for GUI layout. Object-Oriented and Internet-Based Technologies, pp. 110–121 (2004)
5. BOC Group: BPMS (Business Process Management System). http://www.boc-group.com/products/adonis/bpms-method-life-cycle/ (2015). Accessed 18 Sept 2015
6. BOC Group: Business Process Management with ADONIS and the ADONIS Process Portal. http://www.boc-group.com/products/adonis/ (2015). Accessed 18 Sept 2015
7. Börger, E., Stärk, R.: Abstract State Machines: A Method for High-Level System Design and Analysis. Springer, Berlin (2003)
8. Bundesministerium für Bildung und Forschung: Zukunftsprojekt Industrie 4.0. http://www.bmbf.de/de/9072.php (2015). Accessed 30 Sept 2015
9. Davis, R., Brabander, E.: ARIS Design Platform: Getting Started with BPM. Springer, Berlin (2007)
10. Draheim, D., Geist, V., Natschläger, C.: Integrated framework for seamless modeling of business and technical aspects in process-oriented enterprise applications. Int. J. Softw. Eng. Knowl. Eng. **22**(05), 645–674 (2012)
11. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of Business Process Management. Springer, Berlin (2013)
12. Fleischmann, A., Schmidt, W., Stary, C., Obermeier, S., Börger, E.: Subject-Oriented Business Process Management. Springer, Berlin (2012)

13. International Controller Association (ICV): Industrie 4.0 - Controlling in the Age of Intelligent Networks. http://www.icv-controlling.com/fileadmin/Assets/Content/AK/Ideenwerkstatt/Files/Dream_Car_Industrie_4.0_EN.pdf (2015). Accessed 23 Dec 2015
14. International Organization for Standardization: Information technology – Object Management Group: Business Process Model and Notation. http://www.omg.org/spec/BPMN/ISO/19510/PDF/ (2013). Accessed 28 Sept 2015. iSO/IEC 19510
15. Kagermann, P., Wahlster, P., Helbig, D.: Recommendations for implementing the strategic initiative INDUSTRIE 4.0 (2013). Accessed 23 Dec 2015
16. Keller, G., Nüttgens, M., Scheer, A.W.: Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK). http://www.econbiz.de/archiv/sb/usb/iwi/prozessmodellierung_grundlage_prozessketten.pdf (1992). Accessed 08 Oct 2015
17. Kirchmer, M., Franz, P.: BPM-discipline framework. http://bpm-d.com/framework (2015). Accessed 28 Sept 2015
18. Kleinschmidt, T.: Integration von BPMN und UML: Wie Mars und Venus oder doch zwei Seiten einer Medaille? OBJEKTspektrum **04**, 60–65 (2015)
19. Kossak, F., Geist, V.: An enhanced communication concept for business processes. In: Kolb, J., Leopold, H., Mendling, J. (eds.) Enterprise Modelling and Information Systems Architectures – Proceedings of EMISA 2015. Lecture Notes in Informatics, vol. 248, pp. 77–91. Gesellschaft für Informatik (2015)
20. Kossak, F., Illibauer, C., Geist, V., Kubovy, J., Natschläger, C., Ziebermayr, T., Kopetzky, T., Freudenthaler, B., Schewe, K.D.: A Rigorous Semantics for BPMN 2.0 Process Diagrams. Springer, Berlin (2015)
21. Lawlor, B.: The Age of Globalization: Impact of Information Technology on Global Business Strategies. Honors Projects in Computer Information Systems. Paper 1. http://digitalcommons.bryant.edu/honors_cis/1 (2007). Accessed 08 Oct 2015
22. Metasonic: Modeling with S-BPM. https://www.metasonic.de/en/modeling-with-s-bpm (2015). Accessed 08 Oct 2015
23. Metasonic: What's S-BPM. https://www.metasonic.de/en/s-bpm (2015). Accessed 08 Oct 2015
24. Meyer, A., Smirnov, S., Weske, M.: Data in business processes. EMISA Forum **31**(3), 5–31 (2011)
25. Milner, R.: A Calculus of Communicating Systems. Springer, Secaucus (1982)
26. Monostori, L.: Cyber-physical production systems: roots, expectations and R&D challenges. Procedia CIRP **17**, 9–13 (2014)
27. Murata, T.: Petri nets: properties, analysis and applications. Proc. IEEE **77**(4), 541–580 (1989)
28. Natschläger, C., Geist, V.: A layered approach for actor modelling in business processes. Bus. Process Manag. J. **19**, 917–932 (2013)
29. Object Management Group: Business Process Model and Notation (BPMN) 2.0. http://www.omg.org/spec/BPMN/2.0 (2011). Accessed 06 Oct 2015
30. Object Management Group: OMG Systems Modelling Language (OMG SysML). http://www.omg.org/spec/SysML (2015). Accessed 23 Sept 2015
31. Object Management Group: OMG Unified Modeling Language (OMG UML), version 2.5. http://www.omg.org/spec/UML/2.5 (2015). Accessed 06 Oct 2015
32. Oppl, S., Fleischmann, A.: S-BPM ONE - education and industrial developments. In: 4th International Conference, S-BPM ONE 2012, Vienna, Austria, 4–5 April 2012. Proceedings. Communications in Computer and Information Science. Springer, Berlin Heidelberg (2012)
33. Perisic, B., Milosavljevic, G., Dejanovic, I., Milosavljevic, B.: UML profile for specifying user interfaces of business applications. Comput. Sci. Inf. Syst. **8**(2), 405–426 (2011)
34. Petri, C.A.: Kommunikation mit Automaten. Ph.D. thesis, Universität Hamburg (1962)
35. Petri, C.A.: Communication with automata. Rome Air Development Center (New York) TR-65-377 / DTIC Research Report AD0630125 (1966)
36. Pinheiro da Silva, P., Paton, N.: User interface modelling with UML. In: Proceedings of the 10th European-Japanese Conference on Information Modelling and Knowledge Bases (2000)

37. Rauf, A., Rahim, M., Ramzan, M., Shahid, A.: Extending UML to model GUI: a new profile. In: 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE), vol. 1, pp. 349–353 (2010)
38. Rittgen, P.: Enterprise Modeling and Computing with UML. ITPro collection, Idea Group Pub (2006)
39. Scheer, A.W.: ARIS - Business Process Modeling. Springer, Berlin (2000)
40. Scheer, A., Abolhassan, F., Jost, W., Kirchmer, M.: Business Process Excellence: ARIS in Practice. Springer, Berlin (2012)
41. Schewe, K.D.: Horizontal and vertical business process model integration. In: Decker, H., Lenka, L., Link, S., Basl, J., Tjoa, A. (eds.) Database and Expert Systems Applications. Lecture Notes in Computer Science, vol. 8055, pp. 1–3. Springer, Berlin (2013)
42. Schuster, T.: Modellierung. Integration und Analyse von Ressourcen in Geschäftsprozessen, Karlsruher Institut für Technologie (2014)
43. Schönthaler, F., Vossen, G., Oberweis, A., Karle, T.: Geschäftsprozesse für Business Communities. Oldenbourg Verlag München (2011)
44. van der Aalst, W.M.P.: The application of petri nets to workflow management. J. Circuits Syst. Comput. **8**(1), 21–66 (1998)
45. ter Hofstede, A.M., van der Aalst, W.M.P., Adams, M., Russell, N. (eds.): Modern Business Process Automation: YAWL and its Support Environment. Springer, Heidelberg (2010)
46. van der Aalst, W.M.P., van Hee, K.M., ter Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of workflow nets: classification, decidability, and analysis. Form. Asp. Comput. **23**(3), 333–363 (2011)
47. van Hee, K.M., Oanea, O., Sidorova, N.: Colored petri nets to verify extended event-driven process chains. In: Meersman, R., Tari, Z., Hacid, M.S., Mylopoulos, J., Pernici, B., Babaoglu, Z., Jacobsen, H.A., Loyall, J.P., Kifer, M., Spaccapietra, S. (eds.) OTM Conferences (1). Lecture Notes in Computer Science, vol. 3760, pp. 183–201. Springer, Berlin (2005)
48. Wakayama, T., Kannapan, S., Khoong, C., Navathe, S., Yates, J.: Information and Process Integration in Enterprises: Rethinking Documents. The Springer International Series in Engineering and Computer Science, Springer (2012)
49. Wohed, P., van der Aalst, W., Dumas, M., ter Hofstede, A., Russell, N.: On the suitability of BPMN for business process modelling. In: Dustdar, S., Fiadeiro, J., Sheth, A. (eds.) Business Process Management. Lecture Notes in Computer Science, vol. 4102, pp. 161–176. Springer, Berlin (2006)

Chapter 2
Deontic Process Diagrams

In this chapter, we present a deontic classification of tasks, that is, we classify tasks as *obligatory*, *permissible*, *forbidden*, or *alternative* with respect to others. Such a classification can also be made conditional, and we will resort to such conditional deontic modality in our concept of actor modelling (a particular weakness of many Business Process Modelling Languages (BPML)) in Chap. 3.

For such a classification, we employ (modal) deontic logic, a family of modal logics which were originally developed for reasoning on ethical theories (and laws). The name is derived from classical Greek, "deon", which means "obligation"; and obligation has been cho sen in most deontic logics as the most basic modal operator.

In the following, we present a notation for deontically classified tasks, combining symbols and colour encoding, demonstrate their usage based on the *Control-Flow Patterns* (part of the *Workflow Patterns* introduced by van der Aalst et al. [32]), provide a semantics for such tasks, and a model transformation from classical process diagrams to deontic process diagrams. We also show that such a transformation is reliable, or "trusted". An approach to extend the Business Process Model and Notation (BPMN) with deontic logic was suggested by Natschläger [21], which was extended in [23, 24] and subsequently adapted to the Hagenberg Business Process Modelling (H-BPM) method. We consider deontic classification to be optional.

In most BPMLs, deontic modality is only implicitly expressed through the structure of the process flow, but it cannot be seen when just looking at the task in question. In such languages, all activities are (tacitly) obligatory, and whenever something should be optional, a gateway is used to split the process flow, which offers the possibility to execute the activity or to do nothing (expressed by an empty, alternative path). This requires additional modelling elements to split and merge the process flow and a comprehensive understanding of the whole process to identify obligatory, permissible, and alternative activities.

Deontic process diagrams improve readability by highlighting obligatory, permissible, and alternative activities. This further allows reducing the number of gateways and sequence flows and consequently, the structural complexity of the diagram. An algebraic graph transformation from BPMN to Deontic BPMN was suggested by Natschläger [21] and shown to be trusted (reliable) in [23]. According to

F. Kossak et al., *Hagenberg Business Process Modelling Method*,
DOI 10.1007/978-3-319-30496-0_2

Varró et al. [34], the most important correctness properties of a trusted model transformation are termination, uniqueness (confluence), and behaviour preservation. We adapt the graph transformation to H-BPM and show that the transformation from classical process diagrams to deontic process diagrams is also terminating and confluent, which implies global determinism, and that the original diagram and the deontic diagram are semantically equivalent.

This chapter is structured as follows: Sect. 2.1 provides the motivation for extending process diagrams with deontic logic and Sect. 2.2 on related work will focus on deontic logic in process modelling. Afterwards, Sect. 2.3 introduces possible deontic classifications of an activity and provides a discussion of user decisions and conditional decisions and an overview to which extent deontic concepts can be used in process flows. Then the *Workflow Patterns* are presented, which describe the main constructs of BPMLs. Extended with deontic logic are the *Control-Flow Patterns*, which specify the process flow (see Sect. 2.4). The benefits of deontic process diagrams are further demonstrated by an application scenario study (see Sect. 2.5) and Sect. 2.6 provides a semantic model for deontic activities. Finally, we introduce in Sect. 2.7 a graph transformation from classical process diagrams to deontic process diagrams and show that this transformation is trusted, i.e. confluent, terminating, and preserving semantics. We sum up the results in Sect. 2.8.

2.1 Motivation

In standard process diagrams, all activities are implicitly obligatory, and whenever an activity should be optional, the process flow is split so one can choose to do nothing by means of an alternative, empty path (i.e. a path without an activity). This implies that the decision whether to execute one or more activities is described within another element, that is, the splitting element (e.g. a gateway). The separation between decision and execution leads to the following problems:

- Limited readability: It is difficult to identify obligatory, permissible, and alternative activities at first sight.
- Complex structure: Additional elements are necessary to express modality, which complicates the structure of the diagram.
- Duplication: Various agents with different normative concepts for the same activity require a duplication of the activity.

Regarding the last issue, in most BPMLs, activities have to be duplicated if modality depends on a particular role. For example, a task *Attend Conference* might be obligatory for the main author of a paper but permissible for reviewers and co-authors.

Problems with implicit modality were also observed by the authors in an industrial project where an order execution process and a business travel process contained several permissible activities (see Sect. 2.5). The resulting process diagrams were complex and a distinction between obligatory and permissible activities would have been advantageous.

Thus we extend process diagrams with deontic logic, which allows to highlight the modality and to reduce the structural complexity of the process flow. Decisions are specified within the corresponding activities, which permits a fine-granular capture of rights and obligations.

2.2 Related Work

An overview of the main applications of deontic logic in computer science is given by Wieringa and Meyer [37]. Broersen et al. identified ten problems of deontic logic and normative reasoning in computer science [6]. Amongst these problems is how to combine legal ontologies, normative systems, business process notations, and compliance checking tools. For this purpose, the authors recommend the Semantics of Business Vocabulary and Rules (SBVR) for interaction between norms and business processes. SBVR employs alethic and deontic modal logics (see [25]), but it considers neither the influence on the process flow (such as readability or reduction of structural complexity) nor the model transformation.

According to Goedertier and Vanthienen [12], most process modelling languages, such as UML Activity Diagrams (UML ADs), BPMN and the Business Process Execution Language (BPEL), are procedural and only implicitly keep track of why design choices have been made. In contrast, these authors present a vocabulary for declarative process modelling that supports business concerns, execution scenarios, execution mechanisms, modality, rule enforcement, and communication. Considering modality, procedural modelling only specifies what *must* be the case while declarative process modelling supports the modalities *must*, *ought*, and *can* based on deontic logic. In [11], the same authors introduce a language to express temporal rules about obligations and permissions in business interaction, called *Penelope*. Those publications provide a good foundation for the suggested deontic extension of process diagrams. However, the focus of the normative concepts is more on agents and temporal constraints, whereas neither optimisation capabilities nor model transformation are studied.

Other publications focus on formal models of normative reasoning and deontic logic in combination with business rules and process modelling. Padmanabhan et al. [26] consider process modelling and deontic logic. They develop a logical framework based on multi-modal logic to capture the normative positions among agents in an organisational setting. Governatori and Milosevic present a language for expressing contract conditions in terms of deontic concepts, called *Business Contract Language (BCL)* [13]. However, these publications focus on agents and their contractual relationships and do not provide a detailed study of modality.

Further approaches ensure business process compliance based on deontic logic. According to Sadiq et al., process and control modelling are two distinct specifications, but convergence is necessary to achieve business practices that are compliant